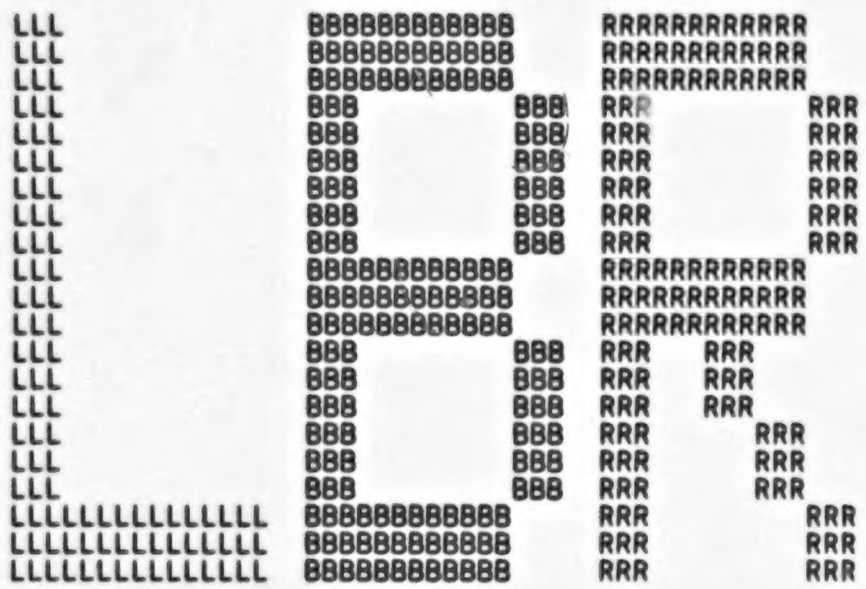

_$2

Val

002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
00F
00F
00F
00F
00F
00F
00F
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF

FILEID**GETHELP

```
  GGGGGGGG  EEEEEEEEEE  TTTTTTTTTT  HH      HH  EEEEEEEEEE  LL          PPPPPPPP
  GGGGGGGG  EEEEEEEEEE  TTTTTTTTTT  HH      HH  EEEEEEEEEE  LL          PPPPPPPP
GG          EE              TT      HH      HH  EE          LL          PP      PP
GG          EE              TT      HH      HH  EE          LL          PP      PP
GG          EE              TT      HH      HH  EE          LL          PP      PP
GG          EE              TT      HH      HH  EE          LL          PP      PP
GG          EEEEEEEE        TT      HHHHHHHHHH  EEEEEEEE    LL          PPPPPPPP
GG          EEEEEEEE        TT      HHHHHHHHHH  EEEEEEEE    LL          PPPPPPPP
GG  GGGGGG  EE              TT      HH      HH  EE          LL          PP
GG  GGGGGG  EE              TT      HH      HH  EE          LL          PP
GG      GG  EE              TT      HH      HH  EE          LL          PP              ....
GG      GG  EE              TT      HH      HH  EE          LL          PP              ....
  GGGGGG    EEEEEEEEEE      TT      HH      HH  EEEEEEEEEE  LLLLLLLLLL  PP              ....
  GGGGGG    EEEEEEEEEE      TT      HH      HH  EEEEEEEEEE  LLLLLLLLLL  PP              ....

LL              IIIIII      SSSSSSSS
LL              IIIIII      SSSSSSSS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II        SSSSSS
LL                II        SSSSSS
LL                II              SS
LL                II              SS
LL                II              SS
LL                II              SS
LLLLLLLLLL      IIIIII    SSSSSSSS
LLLLLLLLLL      IIIIII    SSSSSSSS
```

```
;     1          0001  0 MODULE lbr_gethelp (                        ! Routine to extract help from library
;     2          0002  0                 LANGUAGE (BLISS32),
;     3          0003  0                 IDENT = 'V04-000'
;     4          0004  0                 ) =
;     5          0005  1 BEGIN
;     6          0006  1 %TITLE  'Extract help text from library';
;     7          0007  1 !
;     8          0008  1 !*****************************************************************************
;     9          0009  1 !*                                                                           *
;    10          0010  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;    11          0011  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;    12          0012  1 !*  ALL RIGHTS RESERVED.                                                      *
;    13          0013  1 !*                                                                           *
;    14          0014  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
;    15          0015  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE     *
;    16          0016  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER      *
;    17          0017  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
;    18          0018  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
;    19          0019  1 !*  TRANSFERRED.                                                              *
;    20          0020  1 !*                                                                           *
;    21          0021  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
;    22          0022  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT     *
;    23          0023  1 !*  CORPORATION.                                                              *
;    24          0024  1 !*                                                                           *
;    25          0025  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
;    26          0026  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
;    27          0027  1 !*                                                                           *
;    28          0028  1 !*                                                                           *
;    29          0029  1 !*****************************************************************************
;    30          0030  1
;    31          0031  1 !++
;    32          0032  1 !
;    33          0033  1 ! FACILITY:  Library access procedures
;    34          0034  1 !
;    35          0035  1 ! ABSTRACT:
;    36          0036  1 !
;    37          0037  1 !       The VAX/VMS librarian procedures implement a standard access method
;    38          0038  1 !       to libraries through a shared, common procedure set.
;    39          0039  1 !
;    40          0040  1 ! ENVIRONMENT:
;    41          0041  1 !
;    42          0042  1 !       VAX native, user mode.
;    43          0043  1 !
;    44          0044  1 !--
;    45          0045  1 !
;    46          0046  1 !
;    47          0047  1 ! AUTHOR:  Benn Schreiber,       CREATION DATE:  17-Sep-1979
;    48          0048  1 !
;    49          0049  1 ! MODIFIED BY:
;    50          0050  1 !
;    51          0051  1 !       V03-016 GJA0069         Greg Awdziewicz         28-Feb-1984
;    52          0052  1 !               - Allow more characters in help keys in Scan_Word.
;    53          0053  1 !               - Check validity of first character in help key in
;    54          0054  1 !               Scan_Word.
;    55          0055  1 !
;    56          0056  1 !       V03-015 MCN0140         Maria del C. Nasr       16-Nov-1983
;    57          0057  1 !               Make sure that the key being looked up is not longer
```

```
;    58          0058  1 !                  than the maximum size allowed for the given library.
;    59          0059  1 !
;    60          0060  1 !        V03-014 JWT0114         Jim Teague              20-Apr-1983
;    61          0061  1 !                Activate DCXSHR dynamically when needed.
;    62          0062  1 !
;    63          0063  1 !        V03-013 JWT0098         Jim Teague              01-Mar-1983
;    64          0064  1 !                Clear hlp$v_otherinfo bit on exit
;    65          0065  1 !                from print_options.
;    66          0066  1 !
;    67          0067  1 !        V03-012 JWT0089         Jim Teague              13-Jan-1983
;    68          0068  1 !                Clear up 9th level HELP problem.
;    69          0069  1 !
;    70          0070  1 !        V03-011 JWT0070         Jim Teague              29-Nov-1982
;    71          0071  1 !                Adjustment to previous fix.
;    72          0072  1 !
;    73          0073  1 !        V03-010 JWT0064         Jim Teague              11-Nov-1982
;    74          0074  1 !                Expanded area allocated for DCX records.
;    75          0075  1 !
;    76          0076  1 !        V03-009 JWT0062         Jim Teague              09-Nov-1982
;    77          0077  1 !                Made DCX compress/expand descriptors static.
;    78          0078  1 !
;    79          0079  1 !        V03-008 JWT0056         Jim Teague              17-Sep-1982
;    80          0080  1 !                Equipped lbr$get_help with DCX expansion interface.
;    81          0081  1 !
;    82          0082  1 !        V03-007         RPG49043        Bob Grosso      07-Sep-1982
;    83          0083  1 !                Line_width of 0 didn't default to 80 as it was supposed to.
;    84          0084  1 !
;    85          0085  1 !--
;    86          0086  1
;    87          0087  1
```

```
;    89          0088  1 %SBTTL  'Declarations';
;    90          0089  1 LIBRARY
;    91          0090  1     'SYS$LIBRARY:STARLET';
;    92          0091  1 REQUIRE
;    93          0092  1     'PREFIX';
;    94          0231  1 REQUIRE
;    95          0232  1     'LBRDEF';
;    96          0823  1
;    97          0824  1 LINKAGE
;    98          0825  1     fmg_match = JSB (REGISTER=2, REGISTER=3,
;    99          0826  1                      REGISTER=4, REGISTER=5) : NOTUSED (10, 11); !Linkage for fmg$match_name
;   100          0827  1
;   101          0828  1 EXTERNAL ROUTINE
;   102          0829  1     lbr$load_dcx,
;   103          0830  1     traverse_keys,                          !Traverse index
;   104          0831  1     lookup_key,                             !Lookup key in index
;   105          0832  1     validate_ctl : JSB_1,                   !Validate control index
;   106          0833  1     get_mem : JSB_2,                        !allocate memory
;   107          0834  1     dealloc_mem : JSB_2,
;   108          0835  1     read_record : JSB_2,                    !Read a text record from library
;   109          0836  1     lib$cvt_dtb    : ADDRESSING_MODE(GENERAL),  !Convert decimal to binary
;   110          0837  1     lib$put_output : ADDRESSING_MODE(GENERAL),  !Write line to SYS$OUTPUT
;   111          0838  1     fmg$match_name : fmg_match;             !Match name with wild chars.
;   112          0839  1
;   113          0840  1 EXTERNAL
;   114          0841  1     dcxshr_address,
;   115          0842  1     dcx_expand_data,
;   116          0843  1     lbr$gl_control : REF BBLOCK;            !Pointer to current library control block
;   117          0844  1
;   118          0845  1 EXTERNAL LITERAL
;   119          0846  1     lbr$_invkey,
;   120          0847  1     lbr$_invnam,
;   121          0848  1     lbr$_normal,
;   122          0849  1     lbr$_nothlplib;
;   123          0850  1
;   124          0851  1 FORWARD ROUTINE
;   125          0852  1     move_key,                               !Copy key name to buffer
;   126          0853  1     call_output,                            !Send line to user routine or lib$put_output
;   127          0854  1     print_blankline,                        !Print a blank line
;   128          0855  1     print_nohelp,                           !Tell that no help was found as specified
;   129          0856  1     print_options,                          !Print help available under current topic
;   130          0857  1     print_helptext,                         !Print help text found in library
;   131          0858  1     print_line,                             !print line
;   132          0859  1     print_keys,                             !Print keys found
;   133          0860  1     is_key_on_line,                         !Check for key line
;   134          0861  1     skip_blanks,                            !Skip blanks on line
;   135          0862  1     scan_word,                              !Scan off a word
;   136          0863  1     make_upper_case,                        !Upcase a name
;   137          0864  1     help_check_mtch,                        !Check entries for matches if wild cards
;   138          0865  1     help_check_prtl,                        !Check entries for partial matches
;   139          0866  1     help_do_key1,                           !Process a key1
;   140          0867  1     expand_it;                              !Common routine to expand data
;   141          0868  1
;   142          0869  1 PSECT OWN = $CODE$;                         !Put own data in code psect since its shareable
;   143          0870  1
;   144          0871  1 OWN
;   145          0872  1     nodocmsg : countedstring ('Sorry, no documentation on '),
```

```
;  146          0873  1     otherinfo : countedstring ('Additional information available:');
```

```
;  148          0874  1 %SBTTL  'Routine get_help';
;  149          0875  1 ROUTINE get_help (helpdata) =
;  150          0876  2 BEGIN
;  151          0877  2
;  152          0878  2 !++
;  153          0879  2 !       This routine does the actual looking up of the first level key for lbr$get_help
;  154          0880  2 !
;  155          0881  2 ! Inputs:
;  156          0882  2 !
;  157          0883  2 !       helpdata        address of help data vector set up by lbr$get_help
;  158          0884  2 !
;  159          0885  2 ! Outputs:
;  160          0886  2 !
;  161          0887  2 !       The help request is processed.
;  162          0888  2 !
;  163          0889  2 !--
;  164          0890  2
;  165          0891  2 MAP
;  166          0892  2     helpdata : REF VECTOR [,LONG];
;  167          0893  2
;  168          0894  2 LOCAL
;  169          0895  2     pmatch,
;  170          0896  2      keyirfa : BBLOCK [rfa$c_length];
;  171          0897  2
;  172          0898  2 BIND
;  173          0899  2     helpinfo = .helpdata [hlp$k_info] : BBLOCK,          !Help info
;  174          0900  2     wildflag = helpinfo [hlp$t_wildflags] : BITVECTOR,   !Bit flag true if key is wild
;  175          0901  2     key1desc = .helpdata [hlp$k_key1desc] : BBLOCK;      !Key 1 descriptor
;  176          0902  2
;  177          0903  2 pmatch = false;
;  178          0904  2 !
;  179          0905  2 ! See if any wild characters present in key name
;  180          0906  2 !
;  181          0907  2 IF NOT CH$FAIL (CH$FIND_CH (.key1desc [dsc$w_length], .key1desc [dsc$a_pointer], %ASCII '*'))
;  182          0908  2     OR NOT CH$FAIL (CH$FIND_CH (.key1desc [dsc$w_length], .key1desc [dsc$a_pointer], %ASCII '%'))
;  183          0909  2
;  184          0910  3         THEN BEGIN
;  185          0911  3             wildflag [0] = true;
;  186          0912  4             perform (traverse_keys (1, help_check_mtch, 0, .helpdata))
;  187          0913  3             END
;  188          0914  3
;  189          0915  2         ELSE
;  190          0916  3             BEGIN
;  191          0917  3             LOCAL
;  192          0918  3                 status;
;  193          0919  3             status = lookup_key (1, key1desc, key1rfa);                 !If key is in library
;  194          0920  3             IF (.status EQL lbr$_invkey) THEN return .status;
;  195          0921  3             IF .status
;  196          0922  3             THEN
;  197          0923  4                 perform (help_do_key1 (key1desc, key1rfa, .helpdata))   ! then process it
;  198          0924  3             ELSE
;  199          0925  4                 BEGIN
;  200          0926  4                 wildflag [0] = true;                                    !Partial match counts as wild.
;  201          0927  4                 pmatch = true;
;  202          0928  4                 perform (traverse_keys (1, help_check_prtl, 0, .helpdata)); ! otherwise see if partial match
;  203          0929  4                 wildflag [0] = false;
;  204          0930  3                 END;
```

```
;  205          0931  2              END;
;  206          0932  2
;  207          0933  2          !
;  208          0934  2          ! Check to make sure we found some help text
;  209          0935  2          !
;  210          0936  2          IF NOT .helpinfo [hlp$v_anyhelp]
;  211          0937  3              THEN BEGIN
;  212          0938  3                   IF .pmatch
;  213          0939  4                   THEN BEGIN
;  214          0940  4                        IF .helpinfo [hlp$l_pmatch] EQL 1                    !If there was exactly 1 partial match
;  215          0941  5                        THEN BEGIN
;  216          0942  5                             wildflag [0] = false;
;  217          0943  5                             help_do_key1 (helpinfo [hlp$b_pmtdesc],         !Find the spot to print options from
;  218          0944  5                                           helpinfo [hlp$b_pmtrfa], .helpdata);
;  219          0945  5                             END
;  220          0946  4                        ELSE helpinfo [hlp$l_lastlevel] = 0;
;  221          0947  4                        END
;  222          0948  4
;  223          0949  3                   ELSE
;  224          0950  4                        IF ( .helpinfo [hlp$l_lastlevel] GTR 0 )             !Back up to last found key
;  225          0951  3                        THEN helpinfo [hlp$l_lastlevel] = .helpinfo [hlp$l_lastlevel] - 1;
;  226          0952  3
;  227          0953  3                   IF NOT .helpinfo [hlp$v_anyhelp]                  !If help still not printed
;  228          0954  3                        THEN perform (print_nohelp (.helpdata));     !Print no help info
;  229          0955  2                   END;
;  230          0956  2
;  231          0957  2          RETURN true
;  232          0958  2
;  233          0959  1 END;                                        ! Of get_help


                                                        .TITLE  LBR_GETHELP Extract help text from library
                                                        .IDENT  \V04-000\

                                                        .PSECT  $CODE$,NOWRT,2

                                                1B  00000 NODOCMSG:
                                                        .BYTE   27
6D 75 63 6F 64 20 6F 6E 20 2C 79 72 72 6F 53  00001      .ASCII  \Sorry, no documentation on \                ;
                  20 6E 6F 20 6E 6F 69 74 61 74 6E 65  00010                                                   ;
                                                21  0001C OTHERINFO:
                                                        .BYTE   33                                             ;
6F 66 6E 69 20 6C 61 6E 6F 69 74 69 64 64 41  0001D      .ASCII  \Additional information available:\          ;
62 61 6C 69 61 76 61 20 6E 6F 69 74 61 6D 72  0002C                                                          ;
                                             3A 65 6C  0003B                                                 ;

                                                        .EXTRN  LBR$LOAD_DCX, TRAVERSE_KEYS
                                                        .EXTRN  LOOKUP_KEY, VALIDATE_CTL
                                                        .EXTRN  GET_MEM, DEALLOC_MEM
                                                        .EXTRN  READ_RECORD, LIB$CVT_DTB
                                                        .EXTRN  LIB$PUT_OUTPUT, FMG$MATCH_NAME
                                                        .EXTRN  DCXSHR_ADDRESS, DCX_EXPAND_DATA
                                                        .EXTRN  LBR$GL_CONTROL, LBR$_INVKEY
                                                        .EXTRN  LBR$_INVNAM, LBR$_NORMAL
                                                        .EXTRN  LBR$_NOTHLPLIB

                                            003C 00000 GET_HELP:
```

```
                                            .WORD   Save R2,R3,R4,R5                        ; 0875
                5E          08  C2 00002    SUBL2   #8, SP                                  ;
                54     04   AC  D0 00005    MOVL    HELPDATA, R4                            ; 0899
                52     04   A4  D0 00009    MOVL    4(R4), R2                               ;
                53     14   A4  D0 0000D    MOVL    20(R4), R3                              ; 0901
                            55  D4 00011    CLRL    PMATCH                                  ; 0903
   04   B3      63          2A  3A 00013    LOCC    #42, (R3), @4(R3)                       ; 0907
                            02  12 00018    BNEQ    1$                                      ;
                            51  D4 0001A    CLRL    R1                                      ;
                            51  D5 0001C 1$: TSTL   R1                                      ;
                            0D  12 0001E    BNEQ    3$                                      ;
   04   B3      63          25  3A 00020    LOCC    #37, (R3), @4(R3)                       ; 0908
                            02  12 00025    BNEQ    2$                                      ;
                            51  D4 00027    CLRL    R1                                      ;
                            51  D5 00029 2$: TSTL   R1                                      ;
                            15  13 0002B    BEQL    4$                                      ;
           44   A2          01  88 0002D 3$: BISB2  #1, 68(R2)                              ; 0911
                            54  DD 00031    PUSHL   R4                                      ; 0912
                            7E  D4 00033    CLRL    -(SP)                                   ;
                   0000V    CF  9F 00035    PUSHAB  HELP_CHECK_MTCH                         ;
                            01  DD 00039    PUSHL   #1                                      ;
        0000G   CF          04  FB 0003B    CALLS   #4, TRAVERSE_KEYS                       ;
                            23  11 00040    BRB     5$                                      ;
                   4008     8F  BB 00042 4$: PUSHR  #^M<R3,SP>                              ; 0919
                            01  DD 00046    PUSHL   #1                                      ;
        0000G   CF          03  FB 00048    CALLS   #3, LOOKUP_KEY                          ;
    00000000G   8F          50  D1 0004D    CMPL    STATUS, #LBR$_INVKEY                    ; 0920
                            6E  13 00054    BEQL    12$                                     ;
                10          50  E9 00056    BLBC    STATUS, 6$                              ; 0921
                            54  DD 00059    PUSHL   R4                                      ; 0923
                       04   AE  9F 0005B    PUSHAB  KEY1RFA                                 ;
                            53  DD 0005E    PUSHL   R3                                      ;
        0000V   CF          03  FB 00060    CALLS   #3, HELP_DO_KEY1                        ;
                1E          50  E8 00065 5$: BLBS   STATUS, 7$                              ;
                                04 00068    RET                                             ;
           44   A2          01  88 00069 6$: BISB2  #1, 68(R2)                              ; 0926
                55          01  D0 0006D    MOVL    #1, PMATCH                              ; 0927
                            54  DD 00070    PUSHL   R4                                      ; 0928
                            7E  D4 00072    CLRL    -(SP)                                   ;
                   0000V    CF  9F 00074    PUSHAB  HELP_CHECK_PRTL                         ;
                            01  DD 00078    PUSHL   #1                                      ;
        0000G   CF          04  FB 0007A    CALLS   #4, TRAVERSE_KEYS                       ;
                42          50  E9 0007F    BLBC    STATUS, 12$                             ;
           44   A2          01  8A 00082    BICB2   #1, 68(R2)                              ; 0929
                37     03   A2  E8 00086 7$: BLBS   3(R2), 11$                              ; 0936
                1E          55  E9 0008A    BLBC    PMATCH, 9$                              ; 0938
                01     2C   A2  D1 0008D    CMPL    44(R2), #1                              ; 0940
                            13  12 00091    BNEQ    8$                                      ;
           44   A2          01  8A 00093    BICB2   #1, 68(R2)                              ; 0942
                            54  DD 00097    PUSHL   R4                                      ; 0944
                       38   A2  9F 00099    PUSHAB  56(R2)                                  ;
                       30   A2  9F 0009C    PUSHAB  48(R2)                                  ; 0943
        0000V   CF          03  FB 0009F    CALLS   #3, HELP_DO_KEY1                        ; 0944
                            0D  11 000A4    BRB     10$                                     ; 0940
                       18   A2  D4 000A6 8$: CLRL   24(R2)                                  ; 0946
                            08  11 000A9    BRB     10$                                     ; 0938
                       18   A2  D5 000AB 9$: TSTL   24(R2)                                  ; 0950
```

```
                                03  15 000AE          BLEQ    10$                                  ;
                          18    A2  D7 000B0          DECL    24(R2)                               ; 0951
               0A         03    A2  E8 000B3 10$:     BLBS    3(R2), 11$                           ; 0953
                                54  DD 000B7          PUSHL   R4                                   ; 0954
        0000V  CF               01  FB 000B9          CALLS   #1, PRINT_NOHELP                     ;
               03               50  E9 000BE          BLBC    STATUS, 12$                          ;
               50               01  D0 000C1 11$:     MOVL    #1, R0                               ; 0957
                                    04 000C4 12$:     RET                                          ; 0959

; Routine Size:  197 bytes,    Routine Base:  $CODE$ + 003E
```

```
;  235          0960  1 %SBTTL  'Routine lbr$get_help';
;  236          0961  1 GLOBAL ROUTINE lbr$get_help (control_index, line_width, user_routine, user_data, keyldesc) =
;  237          0962  2 BEGIN
;  238          0963  2 !++
;  239          0964  2 !
;  240          0965  2 ! FUNCTIONAL DESCRIPTION:
;  241          0966  2 !
;  242          0967  2 !       This routine extracts help text from a help library, optionally
;  243          0968  2 !       indents the output, and then prints the line or calls a supplied
;  244          0969  2 !       routine with a string descriptor.
;  245          0970  2 !
;  246          0971  2 !
;  247          0972  2 ! CALLING SEQUENCE:
;  248          0973  2 !
;  249          0974  2 !       status = LBR$GET_HELP (control_index, [line_width, user_routine,
;  250          0975  2 !                              user_data], keyldesc [,key2desc, ...])
;  251          0976  2 !
;  252          0977  2 ! INPUT PARAMETERS:
;  253          0978  2 !
;  254          0979  2 !       control_index   is the control_index obtained from LBR$INI_CONTROL
;  255          0980  2 !       line_width      is address of longword containing linewidth. (D=80)
;  256          0981  2 !       user_routine    address of user typeout routine
;  257          0982  2 !       user_data       address of user data to pass to user typeout routine
;  258          0983  2 !       keyldesc,...    addresses of string descriptors for keys
;  259          0984  2 !
;  260          0985  2 !
;  261          0986  2 ! IMPLICIT INPUTS:
;  262          0987  2 !
;  263          0988  2 !       The HELP library must be open.
;  264          0989  2 !
;  265          0990  2 ! OUTPUT PARAMETERS:
;  266          0991  2 !
;  267          0992  2 !       NONE
;  268          0993  2 !
;  269          0994  2 ! IMPLICIT OUTPUTS:
;  270          0995  2 !
;  271          0996  2 !       If no user_routine is specified, the help text is printed on SYS$OUTPUT
;  272          0997  2 !       using LIB$PUT_OUTPUT.  If there is a user_routine, it is called for
;  273          0998  2 !       each line of help text found or generated.
;  274          0999  2 !
;  275          1000  2 ! ROUTINE VALUE:
;  276          1001  2 !
;  277          1002  2 !       status  lbr$_normal
;  278          1003  2 !               lbr$_nothlplib  Not help library
;  279          1004  2 !               lbr$_invnam     Too many arguments
;  280          1005  2 !               lbr$_invkey     Key is too long
;  281          1006  2 !
;  282          1007  2 ! SIDE EFFECTS:
;  283          1008  2 !       NONE
;  284          1009  2 !
;  285          1010  2 !--
;  286          1011  2
;  287          1012  2 MAP
;  288          1013  2     keyldesc : REF BBLOCK;
;  289          1014  2
;  290          1015  2 LOCAL
;  291          1016  2     helpdata : BBLOCK [lbr$c_pagesize],                     !A place to copy arg list into                  ;
```

```
;  292          1017  2     foundkeys : BBLOCK [hlp$c_maxkeys * dsc$c_s_bln],          !string descriptors for found keys
;  293          1018  2     keydescriptors : BBLOCK [hlp$c_maxkeys * dsc$c_s_bln],     !String descriptors for keys uppercased
;  294          1019  2     ptr,                                       !Temp pointer
;  295          1020  2     curkeydesc : REF BBLOCK,
;  296          1021  2     status,
;  297          1022  2     helpinfo : BBLOCK [hlp$c_size + hlp$c_maxliswid],
;  298          1023  2     desc : BBLOCK [dsc$c_s_bln],
;  299          1024  2     help_help,
;  300          1025  2     dots;                                      !A string of dots
;  301          1026  2
;  302          1027  2 BUILTIN
;  303          1028  2     ACTUALCOUNT,
;  304          1029  2     NULLPARAMETER;
;  305          1030  2
;  306          1031  2 perform (validate_ctl (..control_index));       !Validate control index
;  307          1032  2 BEGIN
;  308          1033  3     BIND
;  309          1034  3         helpvector = helpdata : VECTOR [,LONG],
;  310          1035  3         wildflag = helpinfo [hlp$t_wildflags] : BITVECTOR,     !Bit flags
;  311          1036  3         mykeydesc = keydescriptors : BBLOCK,                   !Key 1 descriptor to be filled in
;  312          1037  3         context = .lbr$gl_control [lbr$l_ctxptr] : BBLOCK,     !Context block
;  313          1038  3         header  = .lbr$gl_control [lbr$l_hdrptr] : BBLOCK;     !Library header
;  314          1039  3 !
;  315          1040  3 ! Check that library is indeed a help library and that there were
;  316          1041  3 ! not too many arguments supplied.
;  317          1042  3 !
;  318          1043  3     IF .header [lhd$b_type] NEQ lbr$c_typ_hlp           !If library is not help library
;  319          1044  3         THEN RETURN lbr$_nothlplib;
;  320          1045  3
;  321          1046  3     IF ACTUALCOUNT() GTR hlp$c_maxkeys + 4              !If too many args
;  322          1047  3         THEN RETURN lbr$_invnam;                        ! then return error
;  323          1048  3
;  324          1049  3 !
;  325          1050  3 ! If the key is longer than the maximum size for this library, return error
;  326          1051  3 !
;  327          1052  4     BEGIN
;  328          1053  4
;  329          1054  4     BIND
;  330          1055  4         indexdesc = header + lhd$c_idxdesc :  BBLOCK;   ! First index descriptor
;  331          1056  4
;  332          1057  4     IF .key1desc [dsc$w_length] GTR .indexdesc [idd$w_keylen] - 1
;  333          1058  4     THEN
;  334          1059  4         RETURN lbr$_invkey;
;  335          1060  4     END;
;  336          1061  3
;  337          1062  3 !
;  338          1063  3 ! Set up the data list that is passed to all the lower level routines.
;  339          1064  3 !
;  340          1065  3     CH$MOVE( ((ACTUALCOUNT () + 1) * 4), control_index - 4, helpdata); !Copy argument list
;  341          1066  3     CH$FILL (0, hlp$c_maxkeys * dsc$c_s_bln, keydescriptors);
;  342          1067  3     help_help = %ASCII 'HELP';                          !Set up string of 'HELP'
;  343          1068  3 !
;  344          1069  3 ! Zero helpinfo
;  345          1070  3 !
;  346          1071  3     helpvector [hlp$k_info] = helpinfo;                 !Point to the info buffer
;  347          1072  3     CH$FILL (0, hlp$c_size, helpinfo);                  !Zero control information
;  348          1073  3 !
```

```
;  349          1074  3 ! If no KEY1 was specified, or it was null, use "HELP", otherwise, convert keyname
;  350          1075  3 ! given to upper case.
;  351          1076  3 !
;  352          1077  3     IF NULLPARAMETER (hlp$k_key1desc)                   !If its not present
;  353          1078  3         OR .key1desc [dsc$w_length] EQL 0               ! or present and null
;  354          1079  3         OR .key1desc [dsc$a_pointer] EQL 0
;  355          1080  4     THEN BEGIN
;  356          1081  4         helpinfo [hlp$v_helphlp] = true;                !Indicate inserting help key
;  357          1082  4         mykey1desc [dsc$w_length] = 4;
;  358          1083  4         mykey1desc [dsc$a_pointer] = help_help;
;  359          1084  4         END
;  360          1085  4     ELSE BEGIN
;  361          1086  4         helpinfo [hlp$v_helphlp] = false;                   !Indicate not inserting help key
;  362        P 1087  4         perform (get_mem (.key1desc [dsc$w_length],         !Allocate storage for key name
;  363          1088  4                           mykey1desc [dsc$a_pointer]));
;  364          1089  4         make_upper_case (.key1desc, mykey1desc);            !Convert to upper case
;  365          1090  3         END;
;  366          1091  3
;  367          1092  3     helpvector [hlp$k_key1desc] = mykey1desc;           !Change arg list
;  368          1093  3     CH$FILL (0, 8, helpinfo [hlp$t_wildflags]);         !Zero wild key flags
;  369          1094  3     IF NULLPARAMETER (hlp$k_linewidth) OR ..line_width EQL 0
;  370          1095  3     THEN
;  371          1096  3         helpinfo [hlp$l_width] = hlp$c_liswidth         !Use default if none or 0 supplied
;  372          1097  3     ELSE
;  373          1098  3         helpinfo [hlp$l_width] = MIN (..line_width, hlp$c_maxliswid);
;  374          1099  3     helpinfo [hlp$l_curptr] = helpinfo + hlp$c_size;
;  375          1100  3     helpinfo [hlp$l_bufdesc] + 4 = .helpinfo [hlp$l_curptr];
;  376          1101  3     helpinfo [hlp$l_bufdesc] = .helpinfo [hlp$l_width];
;  377          1102  3     CH$FILL (0, hlp$c_maxkeys * dsc$c_s_bln, foundkeys); !Zero descriptor array
;  378          1103  3     helpinfo [hlp$l_keylist] = foundkeys;               !Set pointer for lower routines
;  379          1104  3
;  380          1105  3 !
;  381          1106  3 ! See if key1 string contains '...' . If so, flag it and modify the string
;  382          1107  3 ! descriptor to delete it.
;  383          1108  3 !
;  384          1109  3     dots = %ASCII'....';
;  385          1110  3     ptr = CH$FIND_SUB (.mykey1desc [dsc$w_length], .mykey1desc [dsc$a_pointer],
;  386          1111  3         3, dots);
;  387          1112  3     IF NOT CH$FAIL (.ptr)
;  388          1113  4         AND (.ptr EQL (.mykey1desc [dsc$a_pointer] + .mykey1desc[ dsc$w_length] - 3))
;  389          1114  4     THEN BEGIN
;  390          1115  4         helpinfo [hlp$v_allhelp] = true;                    !Flag ... seen
;  391          1116  5         BEGIN
;  392          1117  5             BIND
;  393          1118  5                 wildbits = helpinfo [hlp$t_wildflags] : VECTOR [,LONG];
;  394          1119  5
;  395          1120  5             wildbits [0] = %X 'FFFFFFFE';               !Set all lower keys as wild
;  396          1121  5             wildbits [1] = -1;
;  397          1122  4             END;
;  398          1123  4
;  399          1124  4         mykey1desc [dsc$w_length] = .mykey1desc [dsc$w_length] - 3; ! and adjust key length
;  400          1125  3         END;
;  401          1126  3 !
;  402          1127  3 ! Look at the key descriptors to make sure that no extra, null key descriptors
;  403          1128  3 ! were passed.
;  404          1129  3 !
;  405          1130  3 helpinfo [hlp$l_realkeys] = ACTUALCOUNT () - 4;         !Initially, this is # of keys
```

```
;  406          1131  3
;  407          1132  3 IF .helpinfo [hlp$v_allhelp]                          !If printing all help
;  408          1133  3     OR .helpinfo [hlp$v_helphlp]                      ! or have inserted 'HELP' key
;  409          1134  3     THEN helpinfo [hlp$l_realkeys] = 1;               ! then only look at first key
;  410          1135  3
;  411          1136  3 IF .helpinfo [hlp$l_realkeys] GEQ 2                  !If 2 or more keys
;  412          1137  3 THEN INCRU i FROM 2 TO .helpinfo [hlp$l_realkeys]     ! then look at key2-keyN
;  413          1138  4 DO BEGIN
;  414          1139  4     BIND
;  415          1140  4         keydesc = keydescriptors + dsc$c_s_bln*.i : BBLOCK;
;  416          1141  4
;  417          1142  4     curkeydesc = .helpvector [.i+hlp$k_keyldesc-1];  !Point to next descriptor
;  418          1143  4
;  419          1144  4     IF .curkeydesc EQL 0                            !If 0 descriptor
;  420          1145  4         OR .curkeydesc [dsc$w_length] EQL 0         ! or 0 length
;  421          1146  4         OR .curkeydesc [dsc$a_pointer] EQL 0        ! or 0 pointer
;  422          1147  4         OR CH$FAIL (CH$FIND_NOT_CH                  ! or all blanks
;  423          1148  4             (.curkeydesc [dsc$w_length], .curkeydesc [dsc$a_pointer], %C' '))
;  424          1149  5         THEN BEGIN
;  425          1150  5             helpinfo [hlp$l_realkeys] = .i - 1;     ! Set real number of keys
;  426          1151  5             EXITLOOP;
;  427          1152  5             END
;  428          1153  5         ELSE BEGIN
;  429          1154  5             IF NOT CH$FAIL (CH$FIND_CH (.curkeydesc [dsc$w_length], !Determine if key has wild chars in it
;  430          1155  5                     .curkeydesc [dsc$a_pointer], %ASCII '*'))
;  431          1156  5                 OR NOT CH$FAIL (CH$FIND_CH (.curkeydesc [dsc$w_length],
;  432          1157  5                     .curkeydesc [dsc$a_pointer], %ASCII '%'))
;  433          1158  5                 THEN wildflag [.i-1] = true;
;  434        P 1159  5             perform (get_mem (.curkeydesc [dsc$w_length],   !Allocate memory to hold string
;  435          1160  5                     keydesc [dsc$a_pointer]));
;  436          1161  5             make_upper_case (.curkeydesc, keydesc);         !Convert to upper case
;  437          1162  5             helpvector [.i+hlp$k_keyldesc-1] = keydesc;     !Correct pointer to descriptor in help vecto
;  438          1163  4             END;
;  439          1164  3     END;
;  440          1165  3 !
;  441          1166  3 ! Get the help
;  442          1167  3 !
;  443          1168  3     status = get_help (helpdata);                   !do the help thing
;  444          1169  3 !
;  445          1170  3 ! Deallocate any key strings that were allocated
;  446          1171  3 !
;  447          1172  3     IF NOT .helpinfo [hlp$v_helphlp] THEN           !If keys were present
;  448          1173  3     INCRU i FROM 0 TO hlp$c_maxkeys-1
;  449          1174  4     DO BEGIN
;  450          1175  4         BIND
;  451          1176  4             keydesc = keydescriptors + dsc$c_s_bln*.i : BBLOCK,
;  452          1177  4             curdesc = foundkeys + dsc$c_s_bln*.i : BBLOCK;
;  453          1178  4
;  454          1179  4         IF .curdesc [dsc$w_length] NEQ 0
;  455          1180  4             THEN IF .curdesc [dsc$a_pointer] NEQ 0
;  456          1181  4                 THEN dealloc_mem (.curdesc [dsc$w_length],
;  457          1182  4                                   .curdesc [dsc$a_pointer]);
;  458          1183  4         IF .keydesc [dsc$w_length] NEQ 0
;  459          1184  4             THEN IF .keydesc [dsc$a_pointer] NEQ 0
;  460          1185  4                 THEN dealloc_mem (.keydesc [dsc$w_length],
;  461          1186  4                                   .keydesc [dsc$a_pointer]);
;  462          1187  3         END;
```

```
;  463          1188  2       END;
;  464          1189  2 RETURN .status
;  465          1190  1 END;



                                       0FFC 00000          .ENTRY  LBR$GET_HELP, Save R2,R3,R4,R5,R6,R7,R8,R9,-; 0961
                                                                   R10,R11                                      :
                      5E      FBF4   CE  9E 00002          MOVAB   -1036(SP), SP                                :
                      50        04   BC  D0 00007          MOVL    @CONTROL_INDEX, R0                           ; 1031
                                 0000G   30 0000B          BSBW    VALIDATE_CTL                                 :
                      01            50   E8 0000E          BLBS    STATUS, 1$                                   :
                                         04 00011          RET                                                  :
                      50     0000G   CF  D0 00012 1$:      MOVL    LBR$GL_CONTROL, R0                           ; 1037
                      03        0A   B0  91 00017          CMPB    @10(R0), #3                                  ; 1043
                                     08  13 0001B          BEQL    2$                                           :
                      50 00000000G   8F  D0 0001D          MOVL    #LBR$_NOTHLPLIB, R0                          ; 1044
                                         04 00024          RET                                                  :
                      0E             6C  91 00025 2$:      CMPB    (AP), #14                                    ; 1046
                                     08  1B 00028          BLEQU   3$                                           :
                      50 00000000G   8F  D0 0002A          MOVL    #LBR$_INVNAM, R0                             ; 1047
                                         04 00031          RET                                                  :
              50    0A   A0 000000C4   8F  C1 00032 3$:      ADDL3   #196, 10(R0), R0                             ; 1055
                      56        14   AC  D0 0003B          MOVL    KEY1DESC, R6                                 ; 1057
                      50        02   A0  3C 0003F          MOVZWL  2(R0), R0                                    :
                                     50  D7 00043          DECL    R0                                           :
      50          66  10             00  ED 00045          CMPZV   #0, #16, (R6), R0                            :
                                     08  15 0004A          BLEQ    4$                                           :
                      50 00000000G   8F  D0 0004C          MOVL    #LBR$_INVKEY, R0                             ; 1059
                                         04 00053          RET                                                  :
                      50             6C  9A 00054 4$:      MOVZBL  (AP), R0                                     ; 1065
                      50             04  C4 00057          MULL2   #4, R0                                       :
                      50             04  C0 0005A          ADDL2   #4, R0                                       :
             FE00  CD 6C             50  28 0005D          MOVC3   R0, CONTROL_INDEX-4, HELPDATA                :
 0050  8F         00  6E             00  2C 00063          MOVC5   #0, (SP), #0, #80, KEYDESCRIPTORS            ; 1066
                              016C   CE     0006A                                                               :
                      6E 504C4548    8F  D0 0006D          MOVL    #1347175752, HELP_HELP                       ; 1067
                   FE04  CD    10    AE  9E 00074          MOVAB   HELPINFO, HELPVECTOR+4                       ; 1071
 005C  8F         00  6E             00  2C 0007A          MOVC5   #0, (SP), #0, #92, HELPINFO                  ; 1072
                                10   AE     00081                                                               :
                      05             6C  91 00083          CMPB    (AP), #5                                     ; 1077
                                     0E  1F 00086          BLSSU   5$                                           :
                                14   AC  D5 00088          TSTL    20(AP)                                       :
                                     09  13 0008B          BEQL    5$                                           :
                                     66  B5 0008D          TSTW    (R6)                                         ; 1078
                                     05  13 0008F          BEQL    5$                                           :
                                04   A6  D5 00091          TSTL    4(R6)                                        ; 1079
                                     10  12 00094          BNEQ    6$                                           :
                   13  AE            02  88 00096 5$:      BISB2   #2, HELPINFO+3                               ; 1081
                 016C  CE            04  B0 0009A          MOVW    #4, MYKEY1DESC                               ; 1082
                 0170  CE            6E  9E 0009F          MOVAB   HELP_HELP, MYKEY1DESC+4                      ; 1083
                                     1E  11 000A4          BRB     8$                                           ; 1077
                   13  AE            02  8A 000A6 6$:      BICB2   #2, HELPINFO+3                               ; 1086
                      51      0170   CE  9E 000AA          MOVAB   MYKEY1DESC+4, R1                             ; 1088
                      50             66  3C 000AF          MOVZWL  (R6), R0                                     :
```

```
                                   0000G  30  000B2          BSBW     GET_MEM                                         ;
                    01               50  E8  000B5          BLBS     STATUS, 7$                                      ;
                                         04  000B8          RET                                                      ;
                           016C      CE  9F  000B9  7$:     PUSHAB   MYKEY1DESC                                      ; 1089
                                     56  DD  000BD          PUSHL    R6                                              ;
                   0000V    CF       02  FB  000BF          CALLS    #2, MAKE_UPPER_CASE                             ;
                    FE14    CD  016C CE  9E  000C4  8$:     MOVAB    MYKEY1DESC, HELPVECTOR+20                       ; 1092
       08        00         6E       00  2C  000CB          MOVC5    #0, (SP), #0, #8, HELPINFO+68                   ; 1093
                                 54  AE      000D0                                                                   ;
                    02               6C  91  000D2          CMPB     (AP), #2                                        ; 1094
                                     0A  1F  000D5          BLSSU    9$                                              ;
                                 08  AC  D5  000D7          TSTL     8(AP)                                           ;
                                     05  13  000DA          BEQL     9$                                              ;
                                 08  BC  D5  000DC          TSTL     @LINE_WIDTH                                     ;
                                     07  12  000DF          BNEQ     10$                                             ;
                  30        AE       50  8F  9A  000E1  9$:     MOVZBL   #80, HELPINFO+32                            ; 1096
                                     16  11  000E6          BRB      12$                                             ;
                            50       08  BC  D0  000E8  10$:    MOVL     @LINE_WIDTH, R0                             ; 1098
               00000100     8F           50  D1  000EC          CMPL     R0, #256                                    ;
                                     05  15  000F3          BLEQ     11$                                             ;
                            50     0100  8F  3C  000F5          MOVZWL   #256, R0                                    ;
                  30        AE           50  D0  000FA  11$:    MOVL     R0, HELPINFO+32                             ;
                  1C        AE       6C  AE  9E  000FE  12$:    MOVAB    HELPINFO+92, HELPINFO+12                    ; 1099
                  18        AE       1C  AE  D0  00103          MOVL     HELPINFO+12, HELPINFO+8                     ; 1100
                  14        AE       30  AE  D0  00108          MOVL     HELPINFO+32, HELPINFO+4                     ; 1101
    0050   8F        00     6E           00  2C  0010D          MOVC5    #0, (SP), #0, #80, FOUNDKEYS                ; 1102
                                   01BC  CE      00114                                                               ;
                  34        AE     01BC  CE  9E  00117          MOVAB    FOUNDKEYS, HELPINFO+36                      ; 1103
                  04        AE 2E2E2E2E  8F  D0  0011D          MOVL     #774778414, DOTS                            ; 1109
    0170   DE   016C   CE   04      AE       03  39  00125          MATCHC   #3, DOTS, MYKEY1DESC, @MYKEY1DESC+4    ; 1110
                                     03  13  0012F          BEQL     13$                                             ;
                            53           03  D0  00131          MOVL     #3, R3                                      ;
                            53           03  C2  00134  13$:    SUBL2    #3, R3                                      ;
                                     24  13  00137          BEQL     14$                                             ; 1112
                            50     016C  CE  3C  00139          MOVZWL   MYKEY1DESC, R0                              ; 1113
                            50     0170  CE  C0  0013E          ADDL2    MYKEY1DESC+4, R0                            ;
                            50           03  C2  00143          SUBL2    #3, R0                                      ;
                            50           53  D1  00146          CMPL     PTR, R0                                     ;
                                     12  12  00149          BNEQ     14$                                             ;
                  13        AE       40  8F  88  0014B          BISB2    #64, HELPINFO+3                             ; 1115
                  54        AE           02  CE  00150          MNEGL    #2, WILDBITS                                ; 1120
                  58        AE           01  CE  00154          MNEGL    #1, WILDBITS+4                              ; 1121
                016C        CE           03  A2  00158          SUBW2    #3, MYKEY1DESC                              ; 1124
                  38        AE           6C  9A  0015D  14$:    MOVZBL   (AP), HELPINFO+40                           ; 1130
                  38        AE           04  C2  00161          SUBL2    #4, HELPINFO+40                             ;
             05   13        AE           06  E0  00165          BBS      #6, HELPINFO+3, 15$                         ; 1132
             04   13        AE           01  E1  0016A          BBC      #1, HELPINFO+3, 16$                         ; 1133
                  38        AE           01  D0  0016F  15$:    MOVL     #1, HELPINFO+40                             ; 1134
                            55       38  AE  D0  00173  16$:    MOVL     HELPINFO+40, R5                             ; 1136
                            02           55  D1  00177          CMPL     R5, #2                                      ;
                                     74  19  0017A          BLSS     26$                                             ;
                            52           02  D0  0017C          MOVL     #2, I                                       ; 1137
                                     6A  11  0017F          BRB      25$                                             ;
                            54     016C CE42  7E  00181  17$:    MOVAQ    KEYDESCRIPTORS[I], R4                      ; 1140
                            53     FE10 CD42  D0  00187          MOVL     HELPVECTOR+16[I], CURKEYDESC               ; 1142
                                     16  13  0018D          BEQL     19$                                             ; 1144
                                     63  B5  0018F          TSTW     (CURKEYDESC)                                    ; 1145
```

```
                                          12  13 00191        BEQL    19$                                              ;
                                   04     A3  D5 00193        TSTL    4(CURKEYDESC)                                    ; 1146
                                          0D  13 00196        BEQL    19$                                              ;
        04   B3          63               20  3B 00198        SKPC    #32, (CURKEYDESC), @4(CURKEYDESC)                ; 1148
                                          02  12 0019D        BNEQ    18$                                              ;
                                          51  D4 0019F        CLRL    R1                                               ;
                                          51  D5 001A1  18$:  TSTL    R1                                               ;
                                          07  12 001A3        BNEQ    20$                                              ;
                   38    AE        FF     A2  9E 001A5  19$:  MOVAB   -1(R2), HELPINFO+40                              ; 1150
                                          44  11 001AA        BRB     26$                                              ; 1149
        04   B3          63               2A  3A 001AC  20$:  LOCC    #42, (CURKEYDESC), @4(CURKEYDESC)                ; 1154
                                          02  12 001B1        BNEQ    21$                                              ;
                                          51  D4 001B3        CLRL    R1                                               ;
                                          51  D5 001B5  21$:  TSTL    R1                                               ; 1155
                                          0D  12 001B7        BNEQ    23$                                              ;
        04   B3          63               25  3A 001B9        LOCC    #37, (CURKEYDESC), @4(CURKEYDESC)                ; 1156
                                          02  12 001BE        BNEQ    22$                                              ;
                                          51  D4 001C0        CLRL    R1                                               ;
                                          51  D5 001C2  22$:  TSTL    R1                                               ; 1157
                                          09  13 001C4        BEQL    24$                                              ;
                         50        FF     A2  9E 001C6  23$:  MOVAB   -1(R2), R0                                       ; 1158
             00    54    AE               50  E2 001CA        BBSS    R0, WILDFLAG, 24$                                ;
                         51        04     A4  9E 001CF  24$:  MOVAB   4(R4), R1                                        ; 1160
                         50               63  3C 001D3        MOVZWL  (CURKEYDESC), R0                                 ;
                                       0000G  30 001D6        BSBW    GET_MEM                                          ;
                         63               50  E9 001D9        BLBC    STATUS, 31$                                      ;
                                          18  BB 001DC        PUSHR   #^M<R3,R4>                                       ; 1161
                 0000V   CF               02  FB 001DE        CALLS   #2, MAKE_UPPER_CASE                              ;
                 FE10  CD42               54  D0 001E3        MOVL    R4, HELPVECTOR+16[I]                             ; 1162
                                          52  D6 001E9        INCL    I                                                ; 1137
                         55               52  D1 001EB  25$:  CMPL    I, R5                                            ;
                                          91  1B 001EE        BLEQU   17$                                              ;
                                 FE00     CD  9F 001F0  26$:  PUSHAB  HELPDATA                                         ; 1168
                  FD42   CF               01  FB 001F4        CALLS   #1, GET_HELP                                     ;
                         55               50  D0 001F9        MOVL    R0, STATUS                                       ;
             3B    13    AE               01  E0 001FC        BBS     #1, HELPINFO+3, 30$                              ; 1172
                                          52  D4 00201        CLRL    I                                                ; 1173
                         54  016C  CE42   7E  00203  27$:     MOVAQ   KEYDESCRIPTORS[I], R4                            ; 1176
                         53  018C  CE42   7E  00209           MOVAQ   FOUNDKEYS[I], R3                                 ; 1177
                                          63  B5 0020F        TSTW    (R3)                                             ; 1179
                                          0F  13 00211        BEQL    28$                                              ;
                                   04     A3  D5 00213        TSTL    4(R3)                                            ; 1180
                                          0A  13 00216        BEQL    28$                                              ;
                         51        04     A3  D0 00218        MOVL    4(R3), R1                                        ; 1181
                         50               63  3C 0021C        MOVZWL  (R3), R0                                         ;
                                       0000G  30 0021F        BSBW    DEALLOC_MEM                                      ;
                                          64  B5 00222  28$:  TSTW    (R4)                                             ; 1183
                                          0F  13 00224        BEQL    29$                                              ;
                                   04     A4  D5 00226        TSTL    4(R4)                                            ; 1184
                                          0A  13 00229        BEQL    29$                                              ;
                         51        04     A4  D0 0022B        MOVL    4(R4), R1                                        ; 1185
                         50               64  3C 0022F        MOVZWL  (R4), R0                                         ;
                                       0000G  30 00232        BSBW    DEALLOC_MEM                                      ;
                                          52  D6 00235  29$:  INCL    I                                                ; 1173
                         09               52  D1 00237        CMPL    I, #9                                            ;
                                          C7  1B 0023A        BLEQU   27$                                              ;
                         50               55  D0 0023C  30$:  MOVL    STATUS, R0                                       ; 1189
```

```
                                        04 0023F 31$:     RET                                            ; 1190

; Routine Size:  576 bytes,    Routine Base:  $CODE$ + 0103
```

```
;  467          1191  1 %SBTTL  'Routine help_check_mtch';
;  468          1192  1 ROUTINE help_check_mtch (entry, user_routine, index_desc, helpdata) =
;  469          1193  2 BEGIN
;  470          1194  2 !++
;  471          1195  2 !
;  472          1196  2 ! This routine is called for every entry in the library to see if
;  473          1197  2 ! the entry matches the wild card key descriptor passed to LBR$GET_HELP.
;  474          1198  2 !
;  475          1199  2 ! INPUTS:
;  476          1200  2 !
;  477          1201  2 !       entry           Address of entry descriptor in index
;  478          1202  2 !       user_routine    Not used
;  479          1203  2 !       index_desc      Not used
;  480          1204  2 !       helpdata        Address of data vector created by lbr$get_help
;  481          1205  2 !
;  482          1206  2 ! If the current entry matches the key1 in the help data vector, call
;  483          1207  2 ! help_do_key1 to process it.
;  484          1208  2 !
;  485          1209  2 !--
;  486          1210  2
;  487          1211  2 MAP
;  488          1212  2     entry : REF BBLOCK,
;  489          1213  2     helpdata : REF VECTOR [,LONG],
;  490          1214  2     index_desc : REF BBLOCK;
;  491          1215  2
;  492          1216  2 BIND
;  493          1217  2     helpinfo = .helpdata [hlp$k_info] : BBLOCK,         !Pointer to information structure
;  494          1218  2     keyldesc = helpdata [hlp$k_keyldesc] : REF BBLOCK;  !Start of key descriptor addresses
;  495          1219  2
;  496          1220  2 LOCAL
;  497          1221  2     match_desc : BBLOCK [dsc$c_s_bln],
;  498          1222  2     match_buf : BBLOCK [lbr$c_maxkeylen],
;  499          1223  2     entrydesc : BBLOCK [dsc$c_s_bln];
;  500          1224  2
;  501          1225  2 !
;  502          1226  2 ! Check for wild card match with fmg$match_name
;  503          1227  2 !
;  504          1228  2 entrydesc [dsc$w_length] = .entry [idx$b_keylen];
;  505          1229  2 entrydesc [dsc$a_pointer] = entry [idx$t_keyname];
;  506          1230  2
;  507          1231  2 match_desc [dsc$w_length] = 0;
;  508          1232  2 match_desc [dsc$a_pointer] = match_buf;
;  509          1233  2
;  510          1234  2 make_upper_case ( entrydesc, match_desc );
;  511          1235  2
;  512          1236  2 IF fmg$match_name (.match_desc [dsc$w_length], .match_desc [dsc$a_pointer],
;  513          1237  2                   .keyldesc [dsc$w_length], .keyldesc [dsc$a_pointer])
;  514          1238  2 THEN perform (help_do_key1 (entrydesc, entry [idx$b_rfa], .helpdata));
;  515          1239  2 RETURN true
;  516          1240  1 END;                                        ! Of help_check_mtch


                                03FC 00000 HELP_CHECK_MTCH:
                                           .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9          ; 1192
```

```
                              5E   FF70  CE  9E 00002          MOVAB   -144(SP), SP                          ;
               52     10      AC     14  C1 00007          ADDL3   #20, HELPDATA, R2                     ; 1218
                              56     04  AC  D0 0000C          MOVL    ENTRY, R6                             ; 1228
                              6E     06  A6  9B 00010          MOVZBW  6(R6), ENTRYDESC                      ;
                      04      AE     07  A6  9E 00014          MOVAB   7(R6), ENTRYDESC+4                    ; 1229
                                     F8  AD  B4 00019          CLRW    MATCH_DESC                            ; 1231
                      FC      AD     08  AE  9E 0001C          MOVAB   MATCH_BUF, MATCH_DESC+4               ; 1232
                                     F8  AD  9F 00021          PUSHAB  MATCH_DESC                            ; 1234
                                     04  AE  9F 00024          PUSHAB  ENTRYDESC                             ;
                   0000V      CF         02  FB 00027          CALLS   #2, MAKE_UPPER_CASE                   ;
                              50         62  D0 0002C          MOVL    (R2), R0                              ; 1237
                              55     04  A0  D0 0002F          MOVL    4(R0), R5                             ; 1236
                              54         60  3C 00033          MOVZWL  (R0), R4                              ;
                              53     FC  AD  D0 00036          MOVL    MATCH_DESC+4, R3                      ;
                              52     F8  AD  3C 0003A          MOVZWL  MATCH_DESC, R2                        ;
                                      0000G  30 0003E          BSBW    FMG$MATCH_NAME                        ;
                              10         50  E9 00041          BLBC    R0, 1$                                ;
                                     10  AC  DD 00044          PUSHL   HELPDATA                              ; 1238
                                         56  DD 00047          PUSHL   R6                                    ;
                                     08  AE  9F 00049          PUSHAB  ENTRYDESC                             ;
                   0000V      CF         03  FB 0004C          CALLS   #3, HELP_DO_KEY1                      ;
                              03         50  E9 00051          BLBC    STATUS, 2$                            ;
                              50         01  D0 00054 1$:      MOVL    #1, R0                                ; 1239
                                             04 00057 2$:      RET                                           ; 1240

; Routine Size:  88 bytes,    Routine Base:  $CODE$ + 0343
```

```
;   518          1241  1 %SBTTL  'Routine help_check_prtl';
;   519          1242  1 ROUTINE help_check_prtl (entry, user_routine, index_desc, helpdata) =
;   520          1243  2 BEGIN
;   521          1244  2 !++
;   522          1245  2 !
;   523          1246  2 ! This routine is called for every entry in the index to determine if the
;   524          1247  2 ! entry satisfies a partial match.
;   525          1248  2 !
;   526          1249  2 ! INPUTS:
;   527          1250  2 !
;   528          1251  2 !       entry           address of current entry in the index
;   529          1252  2 !       user_routine    not used
;   530          1253  2 !       index_desc      not used
;   531          1254  2 !       helpdata        address of help data vector set up by lbr$get_help
;   532          1255  2 !
;   533          1256  2 ! The entry is checked for a partial match and help_do_key1 is called
;   534          1257  2 ! if there is a match
;   535          1258  2 !
;   536          1259  2 !--
;   537          1260  2
;   538          1261  2 MAP
;   539          1262  2     entry : REF BBLOCK,
;   540          1263  2     helpdata : REF VECTOR [,LONG];
;   541          1264  2
;   542          1265  2 BIND
;   543          1266  2     helpinfo = .helpdata [hlp$k_info] : BBLOCK,          !Pointer to information structure
;   544          1267  2     keyldesc = helpdata [hlp$k_keyldesc] : REF BBLOCK;  !Start of key descriptor addresses
;   545          1268  2
;   546          1269  2 LOCAL
;   547          1270  2     entrybuf : BBLOCK [lbr$c_maxkeylen],
;   548          1271  2     entrydesc : BBLOCK [dsc$c_s_bln];
;   549          1272  2
;   550          1273  2 entrydesc [dsc$w_length] = .entry [idx$b_keylen];
;   551          1274  2 entrydesc [dsc$a_pointer] = entrybuf;                    ! Temporary store to raise case
;   552          1275  2 CH$MOVE (.entry [idx$b_keylen], entry [idx$t_keyname], entrybuf);
;   553          1276  2 make_upper_case (entrydesc, entrydesc);
;   554          1277  2
;   555          1278  2 IF CH$EQL (.keyldesc [dsc$w_length], entrybuf,          !See if it is a partial match
;   556          1279  2                 .keyldesc [dsc$w_length], .keyldesc [dsc$a_pointer])
;   557          1280  2 THEN
;   558          1281  3     BEGIN
;   559          1282  3     entrydesc [dsc$a_pointer] = entry [idx$t_keyname];
;   560          1283  3     IF (helpinfo [hlp$l_pmatch] = .helpinfo [hlp$l_pmatch] + 1) EQL 1   !If this is first partial match
;   561          1284  4         THEN BEGIN
;   562          1285  4              CH$MOVE (dsc$c_s_bln, entrydesc, helpinfo [hlp$b_pmtdesc]);! then remember descriptor for it
;   563          1286  4              CH$MOVE (rfa$c_length, entry [idx$b_rfa], helpinfo [hlp$b_pmtrfa]);
;   564          1287  3              END;
;   565          1288  3     perform (help_do_key1 (entrydesc, entry [idx$b_rfa], .helpdata));
;   566          1289  2     END;
;   567          1290  2
;   568          1291  2 RETURN true
;   569          1292  1 END;                                    ! Of help_check_partl
```

```
                                       01FC 00000 HELP_CHECK_PRTL:
                                                          .WORD   Save R2,R3,R4,R5,R6,R7,R8                ; 1242
                      5E      FF78  CE  9E 00002          MOVAB   -136(SP), SP                             ;
                      58        10  AC  D0 00007          MOVL    HELPDATA, R8                             ; 1266
                      57        04  A8  D0 0000B          MOVL    4(R8), R7                                ;
                      56        04  AC  D0 0000F          MOVL    ENTRY, R6                                ; 1273
                      6E        06  A6  9B 00013          MOVZBW  6(R6), ENTRYDESC                         ;
                04    AE        08  AE  9E 00017          MOVAB   ENTRYBUF, ENTRYDESC+4                    ; 1274
                      50        06  A6  9A 0001C          MOVZBL  6(R6), R0                                ; 1275
    08    AE    07    A6            50  28 00020          MOVC3   R0, 7(R6), ENTRYBUF                      ;
                                    5E  DD 00026          PUSHL   SP                                       ; 1276
                            04      AE  9F 00028          PUSHAB  ENTRYDESC                                ;
             0000V    CF            02  FB 0002B          CALLS   #2, MAKE_UPPER_CASE                      ;
                      50        14  A8  D0 00030          MOVL    20(R8), R0                               ; 1278
    04    B0    08    AE            60  29 00034          CMPC3   (R0), ENTRYBUF, @4(R0)                   ;
                                    2C  12 0003A          BNEQ    2$                                       ;
                04    AE        07  A6  9E 0003C          MOVAB   7(R6), ENTRYDESC+4                       ; 1282
          50    2C    A7            01  C1 00041          ADDL3   #1, 44(R7), R0                           ; 1283
                2C    A7            50  D0 00046          MOVL    R0, 44(R7)                               ;
                      01            50  D1 0004A          CMPL    R0, #1                                   ;
                                    0A  12 0004D          BNEQ    1$                                       ;
    30    A7          6E            08  28 0004F          MOVC3   #8, ENTRYDESC, 48(R7)                    ; 1285
    38    A7          66            06  28 00054          MOVC3   #6, (R6), 56(R7)                         ; 1286
                          0140      8F  BB 00059 1$:      PUSHR   #^M<R6,R8>                               ; 1288
                            08      AE  9F 0005D          PUSHAB  ENTRYDESC                                ;
             0000V    CF            03  FB 00060          CALLS   #3, HELP_DO_KEY1                         ;
                      03            50  E9 00065          BLBC    STATUS, 3$                               ;
                      50            01  D0 00068 2$:      MOVL    #1, R0                                   ; 1291
                                        04 0006B 3$:      RET                                              ; 1292

; Routine Size:  108 bytes,    Routine Base:  $CODE$ + 039B
```

```
;  571          1293  1 %SBTTL  'Routine move_key';
;  572          1294  1 ROUTINE move_key (helpdata, keydesc, spaces) =
;  573          1295  2 BEGIN
;  574          1296  2 !++
;  575          1297  2 !
;  576          1298  2 ! Copy the key into the buffer
;  577          1299  2 !
;  578          1300  2 ! Inputs:
;  579          1301  2 !
;  580          1302  2 !       helpdata        address of help data vector set up by lbr$get_help
;  581          1303  2 !       keydesc         address of string descriptor for key
;  582          1304  2 !       spaces          number of spaces to leave after key
;  583          1305  2 !
;  584          1306  2 ! Outputs:
;  585          1307  2 !
;  586          1308  2 !       Key is copied into buffer.  New line issued if not enough room.
;  587          1309  2 !
;  588          1310  2 !--
;  589          1311  2
;  590          1312  2 MAP
;  591          1313  2     helpdata : REF VECTOR [,LONG],
;  592          1314  2     keydesc : REF BBLOCK;
;  593          1315  2
;  594          1316  2 LOCAL
;  595          1317  2     newlen;
;  596          1318  2
;  597          1319  2 BIND
;  598          1320  2     helpinfo  = .helpdata [hlp$k_info] : BBLOCK;
;  599          1321  2
;  600          1322  2 newlen = .helpinfo [hlp$l_nchars] + .keydesc [dsc$w_length] + .spaces;
;  601          1323  2 IF .newlen GTRU .helpinfo [hlp$l_width]
;  602          1324  2 THEN
;  603          1325  3     BEGIN
;  604          1326  3     IF .keydesc [dsc$w_length] GTRU .helpinfo [hlp$l_width]
;  605          1327  3     THEN
;  606          1328  4         BEGIN
;  607          1329  4         !
;  608          1330  4         !   The key is too large to fit on a line by itself so wrap it
;  609          1331  4         !   by printing as much as will fit in current buffer, and print
;  610          1332  4         !   rest on the following line.
;  611          1333  4         !
;  612          1334  4         LOCAL
;  613          1335  4             excessdesc : BBLOCK [dsc$c_s_bln],
;  614          1336  4             leftover_len;
;  615          1337  4
;  616          1338  4         leftover_len = .helpinfo [hlp$l_width] - .helpinfo [hlp$l_nchars] - 2;
;  617          1339  4         excessdesc [dsc$w_length] = .keydesc [dsc$w_length] - .leftover_len;
;  618          1340  4         excessdesc [dsc$a_pointer] = .keydesc [dsc$a_pointer] + .leftover_len;
;  619          1341  4         helpinfo [hlp$l_curptr] = CH$MOVE (.leftover_len, .keydesc [dsc$a_pointer],
;  620          1342  4                                            .helpinfo [hlp$l_curptr]);
;  621          1343  4         helpinfo [hlp$l_nchars] = .helpinfo [hlp$l_width];
;  622          1344  4         perform (print_line (.helpdata));
;  623          1345  4         move_key (.helpdata, excessdesc, .spaces);
;  624          1346  4         END
;  625          1347  3     ELSE
;  626          1348  4         BEGIN
;  627          1349  4         perform (print_line (.helpdata));               ! Print what we got
```

```
;   628        1350  4            move_key (.helpdata, .keydesc, .spaces);        ! print what didn't fit on it's own line
;   629        1351  3            END;
;   630        1352  3        END
;   631        1353  2    ELSE
;   632        1354  3        BEGIN
;   633        1355  3        helpinfo [hlp$l_nchars] = .newlen;
;   634        1356  3        helpinfo [hlp$l_curptr] = CH$MOVE (.keydesc [dsc$w_length], .keydesc [dsc$a_pointer],
;   635        1357  3                                  .helpinfo [hlp$l_curptr]) + .spaces;
;   636        1358  2        END;
;   637        1359  2    RETURN true
;   638        1360  1    END;                                    ! Of move_key



                                            00FC 00000 MOVE_KEY:
                                                       .WORD   Save R2,R3,R4,R5,R6,R7                         ; 1294
                         5E          08  C2 00002      SUBL2   #8, SP                                         ;
                         57     04   AC  D0 00005      MOVL    HELPDATA, R7                                   ; 1320
                         56     04   A7  D0 00009      MOVL    4(R7), R6                                      ;
                         52     08   AC  D0 0000D      MOVL    KEYDESC, R2                                    ; 1322
                         50          62  3C 00011      MOVZWL  (R2), R0                                       ;
                         50     10   A6  C0 00014      ADDL2   16(R6), R0                                     ;
                         50     0C   AC  C0 00018      ADDL2   SPACES, NEWLEN                                 ;
                    20   A6          50  D1 0001C      CMPL    NEWLEN, 32(R6)                                 ; 1323
                         53          1B  00020         BLEQU   3$                                             ;
   20  A6       62       10          00  ED 00022      CMPZV   #0, #16, (R2), 32(R6)                          ; 1326
                         34          1B  00028         BLEQU   1$                                             ;
                50  20   A6     10   A6  C3 0002A      SUBL3   16(R6), 32(R6), R0                             ; 1338
                         50          02  C2 00030      SUBL2   #2, LEFTOVER_LEN                               ;
                6E       62          50  A3 00033      SUBW3   LEFTOVER_LEN, (R2), EXCESSDESC                 ; 1339
                    04   AE   04 B240    9E 00037      MOVAB   @4(R2)[LEFTOVER_LEN], EXCESSDESC+4             ; 1340
        0C  B6      04   B2          50  28 0003D      MOVC3   LEFTOVER_LEN, @4(R2), @12(R6)                  ; 1342
                    0C   A6          53  D0 00043      MOVL    R3, 12(R6)                                     ;
                    10   A6     20   A6  D0 00047      MOVL    32(R6), 16(R6)                                 ; 1343
                                     57  DD 0004C      PUSHL   R7                                             ; 1344
                 0000V   CF          01  FB 0004E      CALLS   #1, PRINT_LINE                                 ;
                         32          50  E9 00053      BLBC    STATUS, 5$                                     ;
                                0C   AC  DD 00056      PUSHL   SPACES                                         ; 1345
                                04   AE  9F 00059      PUSHAB  EXCESSDESC                                     ;
                                     0F  11 0005C      BRB     2$                                             ;
                                     57  DD 0005E 1$:  PUSHL   R7                                             ; 1349
                 0000V   CF          01  FB 00060      CALLS   #1, PRINT_LINE                                 ;
                         20          50  E9 00065      BLBC    STATUS, 5$                                     ;
                                0C   AC  DD 00068      PUSHL   SPACES                                         ; 1350
                                     52  DD 0006B      PUSHL   R2                                             ;
                                     57  DD 0006D 2$:  PUSHL   R7                                             ;
                    8D   AF          03  FB 0006F      CALLS   #3, MOVE_KEY                                   ;
                                     10  11 00073      BRB     4$                                             ; 1323
                    10   A6          50  D0 00075 3$:  MOVL    NEWLEN, 16(R6)                                 ; 1355
        0C  B6      04   B2          62  28 00079      MOVC3   (R2), @4(R2), @12(R6)                          ; 1357
                    0C   A6   0C BC43    9E 0007F      MOVAB   @SPACES[R3], 12(R6)                            ;
                         50          01  D0 00085 4$:  MOVL    #1, R0                                         ; 1359
                                         04 00088 5$:  RET                                                    ; 1360

; Routine Size:  137 bytes,    Routine Base:  $CODE$ + 0407
```

```
;  640          1361  1 %SBTTL  'Routine help_do_key1';
;  641          1362  1 ROUTINE help_do_key1 (entrydesc, entryrfa, helpdata) =
;  642          1363  2 BEGIN
;  643          1364  2 !++
;  644          1365  2 ! This routine fully processes help text given the key1 has been looked
;  645          1366  2 ! up successfully.
;  646          1367  2 !
;  647          1368  2 ! Inputs:
;  648          1369  2 !
;  649          1370  2 !       entrydesc       Address of string descriptor for key1
;  650          1371  2 !       entryrfa        Address of rfa for key1
;  651          1372  2 !       helpdata        Address of help data vector set up by lbr$get_help
;  652          1373  2 !
;  653          1374  2 ! Outputs:
;  654          1375  2 !
;  655          1376  2 !       Help information (if any, is output)
;  656          1377  2 !
;  657          1378  2 !--
;  658          1379  2
;  659          1380  2 ROUTINE copy_key (helpdata, desc) =
;  660          1381  3 BEGIN
;  661          1382  3 !++
;  662          1383  3 ! This routine allocates dynamic memory, copies the key name into it,
;  663          1384  3 ! and fills in the appropriate descriptor in the array of descriptors
;  664          1385  3 ! pointed to by helpinfo [hlp$l_keylist].
;  665          1386  3 !
;  666          1387  3 ! Inputs:
;  667          1388  3 !
;  668          1389  3 !       helpdata        Address of help data vector set up by lbr$get_help
;  669          1390  3 !       desc            Address of string descriptor for key
;  670          1391  3 !
;  671          1392  3 ! Outputs:
;  672          1393  3 !
;  673          1394  3 !       memory is allocated and correct descriptor is filled in.
;  674          1395  3 !
;  675          1396  3 !--
;  676          1397  3
;  677          1398  3
;  678          1399  3 MAP
;  679          1400  3     helpdata : REF VECTOR [,LONG],
;  680          1401  3     desc : REF BBLOCK;
;  681          1402  3
;  682          1403  3 BIND
;  683          1404  3     helpinfo = .helpdata [hlp$k_info] : BBLOCK,
;  684          1405  3     keydesc = .helpinfo [hlp$l_keylist]
;  685          1406  3                 + (.helpinfo [hlp$l_curlevel] - 1) * dsc$c_s_bln : BBLOCK;
;  686          1407  3
;  687          1408  3 LOCAL
;  688          1409  3     ptr,
;  689          1410  3     nchars;
;  690          1411  3
;  691          1412  3 nchars = 0;
;  692          1413  3 IF .helpdata [hlp$k_userout] EQL 0
;  693          1414  3     THEN nchars = .helpinfo [hlp$l_curlevel] * hlp$c_keylogtab;
;  694          1415  3
;  695          1416  3 IF .keydesc [dsc$a_pointer] NEQ 0
;  696          1417  3     THEN dealloc_mem (.keydesc [dsc$w_length],                  !Deallocate old string
```

```
;  697          1418  3                                   ,keydesc [dsc$a_pointer]);
;  698          1419  3 perform (get_mem (.desc [dsc$w_length] + .nchars, ptr));          !Allocate memory for string
;  699          1420  3 keydesc [dsc$w_length] = .desc [dsc$w_length] + .nchars;
;  700          1421  3 keydesc [dsc$a_pointer] = .ptr;
;  701          1422  3 IF .nchars NEQ 0                                                   !Pad with spaces if needed
;  702          1423  3     THEN ptr = CH$FILL (%ASCII ' ', .nchars, .ptr);
;  703          1424  3 CH$MOVE (.desc [dsc$w_length], .desc [dsc$a_pointer], .ptr);     !Copy string in
;  704          1425  3 RETURN true
;  705          1426  2 END;                                                             !Of copy_key


                                        0FFC 00000 COPY_KEY:
                                                           .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11          ; 1380
                                             5E    04  C2 00002                       SUBL2   #4, SP
                                             53    04  AC D0 00005                     MOVL    HELPDATA, R3      ; 1404
                                             51    04  A3 D0 00009                     MOVL    4(R3), R1
                                             50    14  A1 D0 0000D                     MOVL    20(R1), R0        ; 1406
                                             52    24 B140 7E 00011                    MOVAQ   @36(R1)[R0], R2
                                             52        08 C2 00016                     SUBL2   #8, R2
                                                       54 D4 00019                     CLRL    NCHARS            ; 1412
                                                   0C  A3 D5 0001B                     TSTL    12(R3)            ; 1413
                                                       05 12 0001E                     BNEQ    1$
                                       54    14  A1       01 78 00020                 ASHL    #1, 20(R1), NCHARS ; 1414
                                                   04  A2 D5 00025 1$:                 TSTL    4(R2)             ; 1416
                                                       0A 13 00028                     BEQL    2$
                                             51    04  A2 D0 0002A                     MOVL    4(R2), R1         ; 1417
                                             50        62 3C 0002E                     MOVZWL  (R2), R0
                                                  0000G 30 00031                     BSBW    DEALLOC_MEM
                                             51        6E 9E 00034 2$:                 MOVAB   PTR, R1           ; 1419
                                             57    08  AC D0 00037                     MOVL    DESC, R7
                                             56        67 3C 0003B                     MOVZWL  (R7), R6
                                       50    56            54 C1 0003E               ADDL3   NCHARS, R6, R0
                                                  0000G 30 00042                     BSBW    GET_MEM
                                             22            50 E9 00045               BLBC    STATUS, 4$
                                       50    56            54 C1 00048               ADDL3   NCHARS, R6, R0    ; 1420
                                             62            50 B0 0004C               MOVW    R0, (R2)
                                                04  A2       6E D0 0004F             MOVL    PTR, 4(R2)        ; 1421
                                                            54 D5 00053             TSTL    NCHARS            ; 1422
                                                            0A 13 00055             BEQL    3$
                                 54    20       6E          00 2C 00057             MOVC5   #0, (SP), #32, NCHARS, @PTR ; 1423
                                                   00  BE       0005C
                                             6E            53 D0 0005E               MOVL    R3, PTR
                                       00    BE    04  B7      56 28 00061 3$:       MOVC3   R6, @4(R7), @PTR  ; 1424
                                             50            01 D0 00067               MOVL    #1, R0            ; 1425
                                                            04 0006A 4$:             RET                       ; 1426

; Routine Size:  107 bytes,    Routine Base:  $CODE$ + 0490
```

```
;  707     1427  2 %SBTTL  'Routine find_help_key';
;  708     1428  2 ROUTINE find_help_key (helpdata, helplevel) =
;  709     1429  2 BEGIN
;  710     1430  2 !++
;  711     1431  2 ! This recursive routine does all the work of finding and printing help text.
;  712     1432  2 !
;  713     1433  2 ! Inputs:
;  714     1434  2 !
;  715     1435  2 !       helpdata        Address of help data vector set up by lbr$get_help
;  716     1436  2 !
;  717     1437  2 !--
;  718     1438  2
;  719     1439  2 MAP
;  720     1440  2     helpdata : REF VECTOR [,LONG];
;  721     1441  2
;  722     1442  2 BIND
;  723     1443  2     header = .lbr$gl_control[lbr$l_hdrptr]: BBLOCK,
;  724     1444  2     helpinfo = .helpdata [hlp$k_info] : BBLOCK,
;  725     1445  2     key2rfa = helpinfo [hlp$b_key2rfa],
;  726     1446  2     wildflag = helpinfo [hlp$t_wildflags] : BITVECTOR;
;  727     1447  2
;  728     1448  2 LOCAL
;  729     1449  2     expand_record,
;  730     1450  2     curkeydesc : REF BBLOCK,
;  731     1451  2     saverfa : BBLOCK [rfa$c_length],
;  732     1452  2     level,
;  733     1453  2     curchar,
;  734     1454  2     helpkey,
;  735     1455  2     qualseen,
;  736     1456  2     is_key,
;  737     1457  2     ch_result,
;  738     1458  2     keylength,
;  739     1459  2     wild_path,
;  740     1460  2     savelastrfa : BBLOCK [rfa$c_length],
;  741     1461  2     lastqualrfa : BBLOCK [rfa$c_length],
;  742     1462  2     token2desc : BBLOCK [dsc$c_s_bln],
;  743     1463  2     tokendesc : BBLOCK [dsc$c_s_bln],
;  744     1464  2     recdesc : BBLOCK [dsc$c_s_bln],
;  745     1465  2     keystring : BBLOCK [hlp$c_maxrecsiz];
;  746     1466  2
;  747     1467  2 IF .header[lhd$l_dcxmapvbn] NEQ 0
;  748     1468  2 THEN
;  749     1469  2     expand_record = true
;  750     1470  2 ELSE
;  751     1471  2     expand_record = false;
;  752     1472  2
;  753     1473  2 IF NOT .helpinfo [hlp$l_readsts]                    !If already at end of file
;  754     1474  2     THEN RETURN true;
;  755     1475  2
;  756     1476  2 !
;  757     1477  2 ! Read records until end of module or exit by finishing
;  758     1478  2 !
;  759     1479  2
;  760     1480  2 qualseen = false;
;  761     1481  2 level = .helplevel;                                !Preset level
;  762     1482  2 helpinfo [hlp$l_lastlevel] = .helplevel;           !Set last level looked at
;  763     1483  2 CH$MOVE (rfa$c_length, helpinfo [hlp$b_lstkeyrfa], !Save last key rfa
```

```
;  764     1484  3                                   savelastrfa);
;  765     1485  3 token2desc [dsc$a_pointer] = keystring;          !preset address part of descriptor
;  766     1486  3
;  767     1487  4 WHILE (
;  768     1488  4         CH$MOVE (rfa$c_length, helpinfo [hlp$b_readrfa], saverfa);
;  769     1489  5         IF (helpinfo [hlp$l_readsts] = read_record (helpinfo [hlp$b_readrfa], recdesc))
;  770     1490  4             AND .expand_record
;  771     1491  4         THEN helpinfo[hlp$l_readsts] = expand_it( recdesc );
;  772     1492  4         .helpinfo[hlp$l_readsts]
;  773     1493  4         )
;  774     1494  4     DO BEGIN
;  775     1495  4
;  776     1496  4         curchar = 0;                                !Preset character
;  777     1497  4         curkeydesc = .helpdata [.helplevel - 1 + hlp$k_key1desc];
;  778     1498  4         IF .helplevel GTR .helpinfo [hlp$l_realkeys]            !If key not really present
;  779     1499  4             THEN curkeydesc = 0;
;  780     1500  4         IF .curkeydesc NEQ 0
;  781     1501  5         THEN BEGIN
;  782     1502  5             curchar = CH$RCHAR (.curkeydesc [dsc$a_pointer]);   !Get 1st char of key
;  783     1503  5             IF .curchar EQL %ASCII '/'                          ! and if its a slash (qualifier)
;  784     1504  5             THEN
;  785     1505  5                 IF .curkeydesc [dsc$w_length] EQL 1             !  and if only one char in name (slash)
;  786     1506  5                 THEN
;  787     1507  6                     BEGIN
;  788     1508  6                     IF .key2rfa EQL 0                                   ! and its the first key this module
;  789     1509  6                     THEN CH$MOVE (rfa$c_length, saverfa, key2rfa);      ! then save it away for printing opt
;  790     1510  6                     EXITLOOP;                                           !   then that's all folks
;  791     1511  6                     END
;  792     1512  5                 ELSE helpinfo [hlp$v_qualhelp] = true;                  !   otherwise flag qualifier help
;  793     1513  4             END;
;  794     1514  4
;  795     1515  5         IF (is_key = is_key_on_line  (helpinfo, recdesc, level, tokendesc)) !If line has a key on it
;  796     1516  4             AND .helpinfo [hlp$v_qualhelp]                      ! and its qualifier help
;  797     1517  4             AND .helpinfo [hlp$v_qualine]                       ! and we found a qualifier line
;  798     1518  4             AND NOT .qualseen                                   ! and we haven't seen a qualifier lately
;  799     1519  5             THEN BEGIN
;  800     1520  5                 CH$MOVE (rfa$c_length, saverfa, lastqualrfa);   !Save RFA of last qualifier
;  801     1521  5                 qualseen = true;                                ! and flag we have seen a qualifier
;  802     1522  4                 END;
;  803     1523  4
;  804     1524  4         IF .is_key
;  805     1525  4             AND .curkeydesc NEQ 0
;  806     1526  5             THEN BEGIN
;  807     1527  5                 keylength = .curkeydesc [dsc$w_length];         !Set length of key
;  808     1528  7                 IF ((.keylength GTR .tokendesc [dsc$w_length])  ! but if key greater than key in text
;  809     1529  6                     AND NOT .wildflag [.helplevel - 1])         ! and this key is not wild
;  810     1530  5                     THEN keylength = 0;                         ! then no match
;  811     1531  5                 END
;  812     1532  4             ELSE keylength = 0;
;  813     1533  4
;  814     1534  4         IF .is_key                                              !If key found on line
;  815     1535  4             AND  .key2rfa EQL 0                                 ! and its the first key this module
;  816     1536  4             THEN CH$MOVE (rfa$c_length, saverfa, key2rfa);      ! then save it away for printing options
;  817     1537  4         ch_result = 1;                                          !Preset for no match
;  818     1538  4         IF .helpinfo [hlp$v_keyline]                            !If we found it on a key line
;  819     1539  4             THEN helpinfo [hlp$v_qualhelp] = false;             ! then make sure we treat as one
;  820     1540  4         IF .is_key                                              !If there is a key on the line
```

```
;  821     1541  5              AND (.helpinfo [hlp$v_allhelp]                          ! and we're doing all help
;  822     1542  6              OR (.level EQL .helplevel                               ! and its the right level
;  823     1543  6              AND make_upper_case (tokendesc, token2desc)             ! (make it upper case)
;  824     1544 10              AND ((((IF .keylength EQL 0
;  825     1545 10                      THEN false
;  826     1546 11                      ELSE (ch_result = CH$COMPARE (.keylength, keystring,
;  827     1547  9                                          .keylength, .curkeydesc [dsc$a_pointer])) EQL 0))
;  828     1548 10              OR (IF (.curchar EQL %ASCII '*'
;  829     1549 10                   AND .helpinfo [hlp$v_qualine])
;  830     1550  9                   OR .keylength EQL 0
;  831     1551  9                   THEN false
;  832     1552  9                   ELSE fmg$match_name (.token2desc [dsc$w_length], keystring,
;  833     1553  5                       .keylength, .curkeydesc [dsc$a_pointer]))))))
;  834     1554  5 !
;  835     1555  5 ! We have a winner, process it
;  836     1556  5 !
;  837     1557  5          THEN BEGIN
;  838     1558  5              recdesc [dsc$w_length] = .recdesc [dsc$w_length] -  !Adjust descriptor
;  839     1559  6                                       (.tokendesc [dsc$a_pointer] - !in case
;  840     1560  5                                       .recdesc [dsc$a_pointer]); !we copy_key it
;  841     1561  5              recdesc [dsc$a_pointer] = .tokendesc [dsc$a_pointer];
;  842     1562  5              IF .ch_result EQL 0                                     !If we got here due to a match
;  843     1563  5                  THEN ch_result = CH$COMPARE (.token2desc [dsc$w_length], keystring, !then check for real mat
;  844     1564  5                                       .keylength, .curkeydesc [dsc$a_pointer]);
;  845     1565  5              CH$MOVE (rfa$c_length, helpinfo [hlp$b_readrfa],        !Save RFA of last found key
;  846     1566  5                                     helpinfo [hlp$b_lstkeyrfa]);
;  847     1567  5              IF NOT .helpinfo [hlp$v_qualhelp]                       !Unless qualifier help
;  848     1568  5                  THEN helpinfo [hlp$[_curlevel] = .level;            ! set help level
;  849     1569  5              wild_path = (.ch_result NEQ 0) OR .helpinfo [hlp$v_allhelp] !Determine if wild key
;  850     1570  5                                     OR .wildflag [.helplevel - 1];
;  851     1571  5 !
;  852     1572  5 ! If this key is on last level, then print the help text
;  853     1573  5 !
;  854     1574  5              IF .level EQL .helpinfo [hlp$l_realkeys]                !If found last key
;  855     1575  5                  OR .helpinfo [hlp$v_allhelp]                        ! or we are printing all help
;  856     1576  6              THEN BEGIN
;  857     1577  6                  IF .helpinfo [hlp$v_qualhelp]                       !If qualifier help
;  858     1578  6                  THEN CH$MOVE (rfa$c_length, lastqualrfa,            ! then set to reread line
;  859     1579  6                                helpinfo [hlp$b_readrfa])
;  860     1580  6                  ELSE perform (copy_key (.helpdata, recdesc));       !Otherwise put on keyname line
;  861     1581  6                  IF .helpinfo [hlp$v_allhelp]                        !If printing all help
;  862     1582  6                      THEN helpinfo [hlp$l_lastlevel] = .level;       ! then set last level correctly
;  863     1583  6                  perform (print_helptext (.helpdata));
;  864     1584  6                  helpinfo [hlp$v_hlpfound] = true;                   !Flag help found this call to help_do_key1
;  865     1585  6                  qualseen = false;                                   !Flag no qualifer seen
;  866     1586  6                          IF NOT .helpinfo [hlp$v_qualhelp]           !Unless qualifier help
;  867     1587  6                              THEN helpinfo [hlp$[_curlevel] = .helpinfo [hlp$l_curlevel]
;  868     1588  6                                                               - 1;
;  869     1589  6                          IF .helpinfo [hlp$l_readsts]                !If last read was not end of file
;  870     1590  7                              THEN BEGIN
;  871   P 1591  7                                  perform (find_help_key (.helpdata,      !  then recurse for next
;  872     1592  7                                                          .helplevel));
;  873     1593  7                                  IF NOT .helpinfo [hlp$l_readsts]
;  874     1594  7                                      THEN EXITLOOP;
;  875     1595  7                                  END
;  876     1596  6                              ELSE EXITLOOP                           !Quit if eom
;  877     1597  6                  END
```

```
;  878          1598  6              ELSE BEGIN
;  879          1599  6                  perform (copy_key (.helpdata, recdesc));          !Put key in buffer
;  880        P 1600  6                  perform (find_help_key (.helpdata, (IF .helpinfo [hlp$v_qualhelp]
;  881        P 1601  6                                                     THEN .helplevel
;  882          1602  6                                                     ELSE .helplevel + 1)));
;  883          1603  6
;  884          1604  6                  IF .helpinfo [hlp$l_readsts]                      !If still more module to go
;  885          1605  7                  THEN BEGIN
;  886          1606  7                      perform (find_help_key (.helpdata, .helplevel)); ! then recurse for more keys
;  887          1607  7                      IF NOT .helpinfo [hlp$l_readsts]              !If we are now at end of module
;  888          1608  7                          THEN EXITLOOP;                            ! then all done
;  889          1609  7                      END
;  890          1610  6                  ELSE EXITLOOP;                                ! exit if at end of module
;  891          1611  5                  END;
;  892          1612  5              END
;  893          1613  5          !
;  894          1614  5          ! Line was not special
;  895          1615  5          !
;  896          1616  5              ELSE BEGIN                                        !
;  897          1617  5                  IF NOT .is_key                                !If no key on line
;  898          1618  6                      OR (.helpinfo [hlp$v_qualhelp]            ! or this is qualifier help
;  899          1619  6                      AND NOT .helpinfo [hlp$v_qualine])        ! and this line not a qualifier line
;  900          1620  5                      THEN qualseen = false;
;  901          1621  5                  IF .is_key                                    !If key on line
;  902          1622  5                      AND .level LSSU .helplevel                ! and its less than level we are looking for
;  903          1623  6                      THEN BEGIN
;  904          1624  6                          CH$MOVE (rfa$c_length, saverfa, helpinfo [hlp$b_readrfa]); !restore rfa of last record
;  905          1625  6                          EXITLOOP;                             !Terminate now
;  906          1626  5                          END;
;  907          1627  4                  END;
;  908          1628  3          END;                                                  !End of WHILE loop
;  909          1629  3
;  910          1630  3          !
;  911          1631  3          ! Make sure some help was found.  If no help was found, and the request is not "..."
;  912          1632  3          ! and no keys above this level were wild, then issue the "no help" message.
;  913          1633  3          !
;  914          1634  3
;  915          1635  4          BEGIN
;  916          1636  4              BUILTIN
;  917          1637  4                  FFS;
;  918          1638  4
;  919          1639  4              LOCAL
;  920          1640  4                  posadr,
;  921          1641  4                  sizadr,
;  922          1642  4                  dstadr;
;  923          1643  4
;  924          1644  4              posadr = 0;                                       !Start at first bit
;  925          1645  4              sizadr = .helplevel - 1;                          !Look at this many bits
;  926          1646  4              wild_path = NOT FFS (posadr, sizadr, wildflag, dstadr);   !Look for a wild key
;  927          1647  3              END;
;  928          1648  3
;  929          1649  3          IF NOT .helpinfo [hlp$v_hlpfound]                     !If no help found
;  930          1650  4              AND NOT (.helpinfo [hlp$v_allhelp]                ! and not "..."
;  931          1651  4                       OR .wild_path)                           ! and not wild path to key
;  932          1652  4              THEN BEGIN
;  933          1653  4                  helpinfo [hlp$v_hlpfound] = true;             !Flag help found this call to do_key1
;  934          1654  4                  helpinfo [hlp$v_anyhelp] = true;              !Flag help found this call to lbr$get_help
```

```
;  935          1655  4           CH$MOVE (rfa$c_length, save'astrfa,                          !Restore last rfa
;  936          1656  4                                      helpinfo [hlp$b_lstkeyrfa]);
;  937          1657  4           perform (print_nohelp (.helpdata));                          !   then print no help available
;  938          1658  3           END;
;  939          1659  3
;  940          1660  3   RETURN true
;  941          1661  2   END;                                                                 !Of find_help_key



                                           0FFC 00000 FIND_HELP_KEY:
                                                      .WORD    Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11                : 1428
                    5E    FF60   CE   9E 00002        MOVAB    -160(SP), SP
                    50   0000G   CF   D0 00007        MOVL     LBR$GL_CONTROL, R0                                  : 1443
                    50      0A   A0   D0 0000C        MOVL     10(R0), R0
      51      04    AC           04   C1 00010        ADDL3    #4, HELPDATA, R1                                    : 1444
                    57           61   D0 00015        MOVL     (R1), R7
                          008C   C0   D5 00018        TSTL     140(R0)                                             : 1467
                                 05   13 0001C        BEQL     1$
                    6E           01   D0 0001E        MOVL     #1, EXPAND_RECORD                                   : 1469
                                 02   11 00021        BRB      2$
                                 6E   D4 00023 1$:    CLRL     EXPAND_RECORD                                       : 1471
              0C    AE      4C   A7   9E 00025 2$:    MOVAB    76(R7), 12(SP)                                      : 1473
                    03      0C   BE   E8 0002A        BLBS     @12(SP), 3$
                              029B   31 0002E        BRW      43$
                            14   AE   D4 00031 3$:    CLRL     QUALSEEN                                            : 1480
                    5A      08   AC   D0 00034        MOVL     HELPLEVEL, R10                                      : 1481
              1C    AE           5A   D0 00038        MOVL     R10, LEVEL
              18    A7           5A   D0 0003C        MOVL     R10, 24(R7)                                         : 1482
F0    AD      56    A7           06   28 00040        MOVC3    #6, 86(R7), SAVELASTRFA                             : 1483
              E4    AD      20   AE   9E 00046        MOVAB    KEYSTRING, TOKEN2DESC+4                             : 1485
              04    AE      50   A7   9E 0004B 4$:    MOVAB    80(R7), 4(SP)                                       : 1488
F8    AD      04    BE           06   28 00050        MOVC3    #6, @4(SP), SAVERFA
                    51      70   AE   9E 00056        MOVAB    RECDESC, R1                                         : 1489
                    50      04   AE   D0 0005A        MOVL     4(SP), R0
                              0000G   30 0005E        BSBW     READ_RECORD
              0C    BE           50   D0 00061        MOVL     R0, @12(SP)
                    0F           50   E9 00065        BLBC     R0, 5$
                    0C           6E   E9 00068        BLBC     EXPAND_RECORD, 5$                                   : 1490
                            70   AE   9F 0006B        PUSHAB   RECDESC                                             : 1491
           0000V    CF           01   FB 0006E        CALLS    #1, EXPAND_IT
              0C    BE           50   D0 00073        MOVL     R0, @12(SP)
                    37      0C   BE   E9 00077 5$:    BLBC     @12(SP), 7$                                         : 1492
                            10   AE   D4 0007B        CLRL     CURCHAR                                             : 1496
      50      04    AC           10   C1 0007E        ADDL3    #16, HELPDATA, R0                                   : 1497
                    58         604A   D0 00083        MOVL     (R0)[R10], CURKEYDESC
              28    A7           5A   D1 00087        CMPL     R10, 40(R7)                                         : 1498
                                 02   15 0008B        BLEQ     6$
                                 58   D4 0008D        CLRL     CURKEYDESC                                          : 1499
                                 59   D4 0008F 6$:    CLRL     R9                                                  : 1500
                                 58   D5 00091        TSTL     CURKEYDESC
                                 24   13 00093        BEQL     9$
                                 59   D6 00095        INCL     R9
              10    AE      04   B8   9A 00097        MOVZBL   @4(CURKEYDESC), CURCHAR                             : 1502
                    2F      10   AE   D1 0009C        CMPL     CURCHAR, #47                                        : 1503
```

```
                                  17   12 000A0          BNEQ    9$                                        ;
                     01           68   B1 000A2          CMPW    (CURKEYDESC), #1                          ; 1505
                                  0E   12 000A5          BNEQ    8$                                        ;
                          3E      A7   D5 000A7          TSTL    62(R7)                                    ; 1508
                                  06   12 000AA          BNEQ    7$                                        ;
         3E    A7     F8   AD     06   28 000AC          MOVC3   #6, SAVERFA, 62(R7)                       ; 1509
                                01E0   31 000B2  7$:     BRW     41$                                       ; 1507
               03     A7          10   88 000B5  8$:     BISB2   #16, 3(R7)                                ; 1512
                          78      AE   9F 000B9  9$:     PUSHAB  TOKENDESC                                 ; 1515
                          20      AE   9F 000BC          PUSHAB  LEVEL                                     ;
                          7B      AC   9F 000BF          PUSHAB  RECDESC                                   ;
                                  57   DD 000C2          PUSHL   R7                                        ;
                  0000V   CF      04   FB 000C4          CALLS   #4, IS_KEY_ON_LINE                        ;
                     18   AE      50   D0 000C9          MOVL    R0, IS_KEY                                ;
                          33  18  AE   E9 000CD          BLBC    IS_KEY, 11$                               ;
         13    03     A7          04   E1 000D1          BBC     #4, 3(R7), 10$                            ; 1516
         0E    03     A7          03   E1 000D6          BBC     #3, 3(R7), 10$                            ; 1517
                          0A  14  AE   E8 000DB          BLBS    QUALSEEN, 10$                             ; 1518
         E8    AD     F8   AD     06   28 000DF          MOVC3   #6, SAVERFA, LASTQUALRFA                  ; 1520
                     14   AE      01   D0 000E5          MOVL    #1, QUALSEEN                              ; 1521
                          17  18  AE   E9 000E9  10$:    BLBC    IS_KEY, 11$                               ; 1524
                          14      59   E9 000ED          BLBC    R9, 11$                                   ; 1525
                          5B      68   3C 000F0          MOVZWL  (CURKEYDESC), KEYLENGTH                   ; 1527
  5B     78    AE         10      00   ED 000F3          CMPZV   #0, #16, TOKENDESC, KEYLENGTH             ; 1528
                                  0B   18 000F9          BGEQ    12$                                       ;
                          52  FF  AA   9E 000FB          MOVAB   -1(R10), R2                               ; 1529
         02    44         A7      52   E0 000FF          BBS     R2, 68(R7), 12$                           ;
                                  5B   D4 00104  11$:    CLRL    KEYLENGTH                                 ; 1532
                     0B       18  AE   E9 00106  12$:    BLBC    IS_KEY, 13$                               ; 1534
                          3E      A7   D5 0010A          TSTL    62(R7)                                    ; 1535
                                  06   12 0010D          BNEQ    13$                                       ;
         3E    A7     F8   AD     06   28 0010F          MOVC3   #6, SAVERFA, 62(R7)                       ; 1536
                     08   AE      01   D0 00115  13$:    MOVL    #1, CH_RESULT                             ; 1537
                          59  02  A7   9E 00119          MOVAB   2(R7), R9                                 ; 1538
               04         69      0A   E1 0011D          BBC     #10, (R9), 14$                            ;
               01         A9      10   8A 00121          BICB2   #16, 1(R9)                                ; 1539
                          03  18  AE   E8 00125  14$:    BLBS    IS_KEY, 15$                               ; 1540
                                0156   31 00129          BRW     39$                                       ;
               57         69      0E   E0 0012C  15$:    BBS     #14, (R9), 22$                            ; 1541
                          5A  1C  AE   D1 00130          CMPL    LEVEL, R10                                ; 1542
                                  03   13 00134          BEQL    17$                                       ;
                                013D   31 00136  16$:    BRW     38$                                       ;
                          E0      AD   9F 00139  17$:    PUSHAB  TOKEN2DESC                                ; 1543
                          7C      AE   9F 0013C          PUSHAB  TOKENDESC                                 ;
                  0000V   CF      02   FB 0013F          CALLS   #2, MAKE_UPPER_CASE                       ;
                          EF      50   E9 00144          BLBC    R0, 16$                                   ;
                                  55   D4 00147          CLRL    R5                                        ; 1544
                                  5B   D5 00149          TSTL    KEYLENGTH                                 ;
                                  04   12 0014B          BNEQ    18$                                       ;
                                  55   D6 0014D          INCL    R5                                        ;
                                  14   11 0014F          BRB     20$                                       ;
                          54      01   D0 00151  18$:    MOVL    #1, R4                                    ; 1546
         04    B8     20   AE     5B   29 00154          CMPC3   KEYLENGTH, KEYSTRING, @4(CURKEYDESC)      ;
                                  03   1A 0015A          BGTRU   19$                                       ;
                          54      01   D9 0015C          SBWC    #1, R4                                    ;
                     08   AE      54   D0 0015F  19$:    MOVL    R4, CH_RESULT                             ;
                                  22   13 00163          BEQL    22$                                       ; 1547
```

```
                          2A      10    AE  D1 00165 20$:     CMPL     CURCHAR, #42                                    ; 1548
                                        04  12 00169          BNEQ     21$                                             ;
               C7         69            0B  E0 0016B          BBS      #11, (R9), 16$                                  ; 1549
                          C4            55  E8 0016F 21$:     BLBS     R5, 16$                                         ; 1550
                          53      20    AE  9E 00172          MOVAB    KEYSTRING, R3                                   ; 1552
                          55      04    A8  D0 00176          MOVL     4(CURKEYDESC), R5                               ;
                          54            5B  D0 0017A          MOVL     KEYLENGTH, R4                                   ;
                          52      E0    AD  3C 0017D          MOVZWL   TOKEN2DESC, R2                                  ;
                               0000G    30 00181              BSBW     FMG$MATCH_NAME                                  ;
                          AF            50  E9 00184          BLBC     R0, 16$                                         ;
               50    74   AE      7C    AE  C3 00187 22$:     SUBL3    TOKENDESC+4, RECDESC+4, R0                      ; 1560
                     70   AE            50  A0 0018D          ADDW2    R0, RECDESC                                     ; 1559
                     74   AE      7C    AE  D0 00191          MOVL     TOKENDESC+4, RECDESC+4                          ; 1561
                                  08    AE  D5 00196          TSTL     CH_RESULT                                       ; 1562
                                        15  12 00199          BNEQ     24$                                             ;
                          54            01  D0 0019B          MOVL     #1, R4                                          ; 1563
      5B       00    20   AE      E0    AD  2D 0019E          CMPC5    TOKEN2DESC, KEYSTRING, #0, KEYLENGTH, -         ;
                                  04    B8     001A5                   @4(CURKEYDESC)                                  ;
                                        03  1A 001A7          BGTRU    23$                                             ;
                          54            01  D9 001A9          SBWC     #1, R4                                          ;
                     08   AE            54  D0 001AC 23$:     MOVL     R4, CH_RESULT                                   ;
           56  A7    04   BE            06  28 001B0 24$:     MOVC3    #6, @4(SP), 86(R7)                              ; 1566
               05         69            0C  E0 001B6          BBS      #12, (R9), 25$                                  ; 1567
                     14   A7      1C    AE  D0 001BA          MOVL     LEVEL, 20(R7)                                   ; 1568
                                        50  D4 001BF 25$:     CLRL     R0                                              ; 1569
                                  08    AE  D5 001C1          TSTL     CH_RESULT                                       ;
                                        02  13 001C4          BEQL     26$                                             ;
                                        50  D6 001C6          INCL     R0                                              ;
      51       69         01            0E  EF 001C8 26$:     EXTZV    #14, #1, (R9), R1                               ;
                          50            51  C8 001CD          BISL2    R1, R0                                          ;
                          52      FF    AA  9E 001D0          MOVAB    -1(R10), R2                                     ; 1570
      51   44  A7         01            52  EF 001D4          EXTZV    R2, #1, 68(R7), R1                              ;
               56         51            50  C9 001DA          BISL3    R0, R1, WILD_PATH                               ;
                     28   A7      1C    AE  D1 001DE          CMPL     LEVEL, 40(R7)                                   ; 1574
                                        04  13 001E3          BEQL     27$                                             ;
               4E         69            0E  E1 001E5          BBC      #14, (R9), 32$                                  ; 1575
               08         69            0C  E1 001E9 27$:     BBC      #12, (R9), 28$                                  ; 1577
           04  BE    E8   AD            06  28 001ED          MOVC3    #6, LASTQUALRFA, @4(SP)                         ; 1579
                                        0E  11 001F3          BRB      29$                                             ;
                                  70    AE  9F 001F5 28$:     PUSHAB   RECDESC                                         ; 1580
                                  04    AC  DD 001F8          PUSHL    HELPDATA                                        ;
                   FD95   CF            02  FB 001FB          CALLS    #2, COPY_KEY                                    ;
                          69            50  E9 00200          BLBC     STATUS, 35$                                     ;
               05         69            0E  E1 00203 29$:     BBC      #14, (R9), 30$                                  ; 1581
                     18   A7      1C    AE  D0 00207          MOVL     LEVEL, 24(R7)                                   ; 1582
                                  04    AC  DD 0020C 30$:     PUSHL    HELPDATA                                        ; 1583
                  0000V   CF            01  FB 0020F          CALLS    #1, PRINT_HELPTEXT                              ;
                          55            50  E9 00214          BLBC     STATUS, 35$                                     ;
                     01   A9            20  88 00217          BISB2    #32, 1(R9)                                      ; 1584
                                  14    AE  D4 0021B          CLRL     QUALSEEN                                        ; 1585
               03         69            0C  E0 0021E          BBS      #12, (R9), 31$                                  ; 1586
                                  14    A7  D7 00222          DECL     20(R7)                                          ; 1588
                          6C      0C    BE  E9 00225 31$:     BLBC     @12(SP), 41$                                    ; 1589
                                        5A  DD 00229          PUSHL    R10                                             ; 1592
                                  04    AC  DD 0022B          PUSHL    HELPDATA                                        ;
                   FDCD   CF            02  FB 0022E          CALLS    #2, FIND_HELP_KEY                               ;
                          39            50  E8 00233          BLBS     STATUS, 36$                                     ;
```

```
                                          04 00236        RET                                             ; 1593
                            70     AE     9F 00237 32$:   PUSHAB  RECDESC                                 ; 1599
                            04     AC     DD 0023A        PUSHL   HELPDATA                                ;
              FD53   CF            02     FB 0023D        CALLS   #2, COPY_KEY                            ;
                     27            50     E9 00242        BLBC    STATUS, 35$                             ;
         04          69            0C     E1 00245        BBC     #12, (R9), 33$                          ; 1602
                                   5A     DD 00249        PUSHL   R10                                     ;
                                   06     11 0024B        BRB     34$                                     ;
                     50     01     AA     9E 0024D 33$:   MOVAB   1(R10), R0                              ;
                                   50     DD 00251        PUSHL   R0                                      ;
                            04     AC     DD 00253 34$:   PUSHL   HELPDATA                                ;
              FDA5   CF            02     FB 00256        CALLS   #2, FIND_HELP_KEY                       ;
                     71            50     E9 0025B        BLBC    STATUS, 44$                             ;
                     33     0C     BE     E9 0025E        BLBC    @12(SP), 41$                            ; 1604
                                   5A     DD 00262        PUSHL   R10                                     ; 1606
                            04     AC     DD 00264        PUSHL   HELPDATA                                ;
              FD94   CF            02     FB 00267        CALLS   #2, FIND_HELP_KEY                       ;
                     60            50     E9 0026C 35$:   BLBC    STATUS, 44$                             ;
                     22     0C     BE     E9 0026F 36$:   BLBC    @12(SP), 41$                            ; 1607
                                 FDD5     31 00273 37$:   BRW     4$                                      ;
                     08     18     AE     E9 00276 38$:   BLBC    IS_KEY, 39$                             ; 1617
         07          69            0C     E1 0027A        BBC     #12, (R9), 40$                          ; 1618
         03          69            0B     E0 0027E        BBS     #11, (R9), 40$                          ; 1619
                            14     AE     D4 00282 39$:   CLRL    QUALSEEN                                ; 1620
                     EA     18     AE     E9 00285 40$:   BLBC    IS_KEY, 37$                             ; 1621
                     5A     1C     AE     D1 00289        CMPL    LEVEL, R10                              ; 1622
                                   E4     1E 0028D        BGEQU   37$                                     ;
    04   BE     F8   AD            06     28 0028F        MOVC3   #6, SAVERFA, @4(SP)                     ; 1624
                                   52     D4 00295 41$:   CLRL    POSADR                                  ; 1644
                     50     FF     AA     9E 00297        MOVAB   -1(R10), SIZADR                         ; 1645
                                   51     D4 0029B        CLRL    R1                                      ; 1646
53  44   A7          50            52     EA 0029D        FFS     POSADR, SIZADR, 68(R7), DSTADR          ;
                                   02     12 002A3        BNEQ    42$                                     ;
                                   51     D6 002A5        INCL    R1                                      ;
                     56            51     D2 002A7 42$:   MCOML   R1, WILD_PATH                           ;
         1D     03   A7            05     E0 002AA        BBS     #5, 3(R7), 43$                          ; 1649
         18     03   A7            06     E0 002AF        BBS     #6, 3(R7), 43$                          ; 1650
                     15            56     E8 002B4        BLBS    WILD_PATH, 43$                          ; 1651
                03   A7            21     88 002B7        BISB2   #33, 3(R7)                              ; 1654
    56   A7     F0   AD            06     28 002BB        MOVC3   #6, SAVELASTRFA, 86(R7)                 ; 1656
                            04     AC     DD 002C1        PUSHL   HELPDATA                                ; 1657
             0000V   CF            01     FB 002C4        CALLS   #1, PRINT_NOHELP                        ;
                     03            50     E9 002C9        BLBC    STATUS, 44$                             ;
                     50            01     D0 002CC 43$:   MOVL    #1, R0                                  ; 1660
                                          04 002CF 44$:   RET                                             ; 1661

; Routine Size:  720 bytes,    Routine Base:  $CODE$ + 04FB
```

```
;  943          1662  2 %SBTTL  'Routine help_do_key1';
;  944          1663  2 !
;  945          1664  2 ! Main body of help_do_key1
;  946          1665  2 !
;  947          1666  2
;  948          1667  2 MAP
;  949          1668  2     helpdata : REF VECTOR [,LONG];
;  950          1669  2
;  951          1670  2 LOCAL
;  952          1671  2     expand_record,
;  953          1672  2     helpkey,
;  954          1673  2     recdesc : BBLOCK [dsc$c_s_bln];
;  955          1674  2
;  956          1675  2 BIND
;  957          1676  2     header = .lbr$gl_control[lbr$l_hdrptr] : BBLOCK,
;  958          1677  2     context = .lbr$gl_control [lbr$l_ctxptr] : BBLOCK,  !Context block
;  959          1678  2     helpinfo = .helpdata [hlp$k_info] : BBLOCK,         !Pointer to information structure
;  960          1679  2     keydesc = helpdata [hlp$k_keydesc] : REF BBLOCK;  !Start of key descriptor addresses
;  961          1680  2
;  962          1681  2 IF .header[lhd$l_dcxmapvbn] NEQ 0
;  963          1682  2 THEN
;  964          1683  2     expand_record = true
;  965          1684  2 ELSE
;  966          1685  2     expand_record = false;
;  967          1686  2
;  968          1687  2 helpinfo [hlp$v_uothinfo] = false;                    !Not doing other info text now
;  969          1688  2 helpinfo [hlp$v_unohlp] = false;                     !Haven't determined if help or not yet
;  970          1689  2 helpinfo [hlp$v_ukeylin] = false;                    !Not a key line
;  971          1690  2 helpinfo [hlp$l_curlevel] = 1;                       !Now at level 1
;  972          1691  2 helpinfo [hlp$l_lastlevel] = 1;                      !Last looked at level 1
;  973          1692  2 helpkey = %ASCII 'HELP';
;  974          1693  2 CH$FILL (0, rfa$c_length, helpinfo [hlp$b_key2rfa]);     !Zero key2 rfa
;  975          1694  2 CH$MOVE (rfa$c_length, .entryrfa, helpinfo [hlp$b_readrfa]); !Copy the RFA
;  976          1695  2 CH$FILL (%ASCII ' ', hlp$c_maxrecsiz, .(helpinfo [hlp$l_bufdesc] + 4));
;  977          1696  2
;  978          1697  2 perform (copy_key (.helpdata, .entrydesc));                !Copy key1 into buffer
;  979          1698  2
;  980          1699  2 !
;  981          1700  2 ! Read and skip module header and first record ("1 KEY1")
;  982          1701  2 !
;  983          1702  2
;  984          1703  3 IF NOT (helpinfo [hlp$l_readsts] = read_record (helpinfo [hlp$b_readrfa], recdesc))     !Read and skip modul
;  985          1704  2    THEN RETURN .helpinfo [hlp$l_readsts];
;  986          1705  4 IF NOT ( IF(helpinfo [hlp$l_readsts] = read_record (helpinfo [hlp$b_readrfa], recdesc))
;  987          1706  3                AND .expand_record
;  988          1707  3           THEN helpinfo[hlp$l_readsts] = expand_it( recdesc );
;  989          1708  3           .helpinfo[hlp$l_readsts] )
;  990          1709  2 THEN RETURN .helpinfo [hlp$l_readsts];
;  991          1710  2 CH$MOVE (rfa$c_length, helpinfo [hlp$b_readrfa], helpinfo [hlp$b_lstkeyrfa]);   !Remember RFA of first good
;  992          1711  2
;  993          1712  2 !
;  994          1713  2 ! If there was only one key on the line then handle that.
;  995          1714  2 !
;  996          1715  2
;  997          1716  2 IF .helpinfo [hlp$l_realkeys] EQL 1                    !If only one key
;  998          1717  2    AND NOT .helpinfo [hlp$v_allhelp]                    ! and not '...'
;  999          1718  3    THEN BEGIN
```

```
; 1000     1719  3          IF NOT .context [ctx$v_outputhlp]                     !If not for LBR$OUTPUT_HELP
; 1001     1720  3             AND CH$EQL (.key1desc [dsc$w_length],              ! and 'HELP' keyword
; 1002     1721  3                         .key1desc [dsc$a_pointer],
; 1003     1722  3                         .key1desc [dsc$w_length],
; 1004     1723  3                         helpkey)
; 1005     1724  3             THEN helpinfo [hlp$v_helphlp] = true;              ! then print additional info
; 1006     1725  3          RETURN print_helptext (.helpdata);                    ! then print text and return
; 1007     1726  3          END
; 1008     1727  3
; 1009     1728  3   !
; 1010     1729  3   ! There is more than 1 key.  Search the help text for the text to print
; 1011     1730  3   !
; 1012     1731  3
; 1013     1732  2   ELSE
; 1014     1733  3       BEGIN
; 1015     1734  3          IF .helpinfo [hlp$v_allhelp]                          !If "..." then print help for key1 also
; 1016     1735  3             THEN perform (print_helptext (.helpdata));
; 1017     1736  3          helpinfo [hlp$v_hlpfound] = false;                    !Flag no help found this call to do_key1
; 1018     1737  3          find_help_key (.helpdata, 2);                         !Find the help text and print it
; 1019     1738  2          END;
; 1020     1739  2
; 1021     1740  2   RETURN true
; 1022     1741  1   END;                                  ! Of help_do_key1


                                                    0FFC 00000 HELP_DO_KEY1:
                                                               .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11        ; 1362
                          5E             0C  C2 00002          SUBL2   #12, SP                                     ;
                          50      0000G  CF  D0 00005          MOVL    LBR$GL_CONTROL, R0                          ; 1676
                          51         0A  A0  D0 0000A          MOVL    10(R0), R1                                  ;
                          59         0E  A0  D0 0000E          MOVL    14(R0), R9                                  ; 1677
                          57         0C  AC  D0 00012          MOVL    HELPDATA, R7                                ; 1678
                          56         04  A7  D0 00016          MOVL    4(R7), R6                                   ;
                                   008C  C1  D5 0001A          TSTL    140(R1)                                     ; 1681
                                         05  13 0001E          BEQL    1$                                          ;
                          58             01  D0 00020          MOVL    #1, EXPAND_RECORD                           ; 1683
                                         02  11 00023          BRB     2$                                          ;
                                         58  D4 00025 1$:      CLRL    EXPAND_RECORD                               ; 1685
                          66             07  8A 00027 2$:      BICB2   #7, (R6)                                    ; 1689
                      14  A6             01  D0 0002A          MOVL    #1, 20(R6)                                  ; 1690
                      18  A6             01  D0 0002E          MOVL    #1, 24(R6)                                  ; 1691
                          6E   504C4548  8F  D0 00032          MOVL    #1347175752, HELPKEY                        ; 1692
      06            00    6E             00  2C 00039          MOVC5   #0, (SP), #0, #6, 62(R6)                    ; 1693
                                     3E  A6     0003E                                                              ;
 0050   8F      50  A6  20       08  BC  06  28 00040          MOVC3   #6, @ENTRYRFA, 80(R6)                       ; 1694
                          6E             00  2C 00046          MOVC5   #0, (SP), #32, #80, @8(R6)                  ; 1695
                                     08  B6     0004D                                                              ;
                                     04  AC  DD 0004F          PUSHL   ENTRYDESC                                   ; 1697
                                         57  DD 00052          PUSHL   R7                                          ;
                  FC6C    CF             02  FB 00054          CALLS   #2, COPY_KEY                                ;
                          73             50  E9 00059          BLBC    STATUS, 9$                                  ;
                          52         4C  A6  9E 0005C          MOVAB   76(R6), R2                                  ; 1703
                          51         04  AE  9E 00060          MOVAB   RECDESC, R1                                 ;
                          50         50  A6  9E 00064          MOVAB   80(R6), R0                                  ;
```

```
                                 0000G  30 00068          BSBW    READ_RECORD                       ;
                      62            50  D0 0006B          MOVL    R0, (R2)                          ;
                      22            50  E9 0006E          BLBC    R0, 4$                            ;
                      51       04   AE  9E 00071          MOVAB   RECDESC, R1                       ; 1705
                      50       50   A6  9E 00075          MOVAB   80(R6), R0                        ;
                                 0000G  30 00079          BSBW    READ_RECORD                       ;
                      62            50  D0 0007C          MOVL    R0, (R2)                          ;
                      0E            50  E9 0007F          BLBC    R0, 3$                            ;
                      0B            58  E9 00082          BLBC    EXPAND_RECORD, 3$                 ; 1706
                               04   AE  9F 00085          PUSHAB  RECDESC                           ; 1707
                 0000V CF           01  FB 00088          CALLS   #1, EXPAND_IT                     ;
                      62            50  D0 0008D          MOVL    R0, (R2)                          ;
                      04            62  E8 00090 3$:      BLBS    (R2), 5$                          ; 1708
                      50            62  D0 00093 4$:      MOVL    (R2), R0                          ; 1709
                                    04     00096          RET                                       ;
    56     A6      50 A6            06  28 00097 5$:      MOVC3   #6, 80(R6), 86(R6)                ; 1710
                      01       28   A6  D1 0009D          CMPL    40(R6), #1                        ; 1716
                      20            12     000A1          BNEQ    7$                                ;
           20      03 A6            06  E0 000A3          BBS     #6, 3(R6), 8$                     ; 1717
                      0F       05   A9  E8 000A8          BLBS    5(R9), 6$                         ; 1719
                      50       14   A7  D0 000AC          MOVL    20(R7), R0                        ; 1720
           6E      04 B0            60  29 000B0          CMPC3   (R0), @4(R0), HELPKEY             ;
                      04            12     000B5          BNEQ    6$                                ;
                   03 A6            02  88 000B7          BISB2   #2, 3(R6)                         ; 1724
                                    57  DD 000BB 6$:      PUSHL   R7                                ; 1725
                 0000V CF           01  FB 000BD          CALLS   #1, PRINT_HELPTEXT                ;
                                    04     000C2          RET                                       ;
           0A      03 A6            06  E1 000C3 7$:      BBC     #6, 3(R6), 10$                    ; 1734
                                    57  DD 000C8 8$:      PUSHL   R7                                ; 1735
                 0000V CF           01  FB 000CA          CALLS   #1, PRINT_HELPTEXT                ;
                      10            50  E9 000CF 9$:      BLBC    STATUS, 11$                       ;
                   03 A6            20  8A 000D2 10$:     BICB2   #32, 3(R6)                        ; 1736
                                    02  DD 000D6          PUSHL   #2                                ; 1737
                                    57  DD 000D8          PUSHL   R7                                ;
                  FC51 CF           02  FB 000DA          CALLS   #2, FIND_HELP_KEY                 ;
                      50            01  D0 000DF          MOVL    #1, R0                            ; 1740
                                    04     000E2 11$:     RET                                       ; 1741

; Routine Size:  227 bytes,    Routine Base:  $CODE$ + 07CB
```

```
;  1024          1742  1 %SBTTL  'Routine print_helptext';
;  1025          1743  1 ROUTINE print_helptext (helpdata) =
;  1026          1744  2 BEGIN
;  1027          1745  2 !++
;  1028          1746  2 ! Print some help text
;  1029          1747  2 !
;  1030          1748  2 ! Inputs:
;  1031          1749  2 !
;  1032          1750  2 !       helpdata        Address of help data vector set up by lbr$get_help
;  1033          1751  2 !
;  1034          1752  2 ! Outputs:
;  1035          1753  2 !
;  1036          1754  2 !       localrfa        updated
;  1037          1755  2 !       help text is output
;  1038          1756  2 !
;  1039          1757  2 !--
;  1040          1758  2
;  1041          1759  2 MAP
;  1042          1760  2     helpdata : REF VECTOR [,LONG];
;  1043          1761  2
;  1044          1762  2 LOCAL
;  1045          1763  2     expand_record,
;  1046          1764  2     dataseen,
;  1047          1765  2     recdesc : BBLOCK [dsc$c_s_bln],
;  1048          1766  2     saverfa : BBLOCK [rfa$c_length],
;  1049          1767  2     level,
;  1050          1768  2     keydesc : BBLOCK [dsc$c_s_bln];
;  1051          1769  2
;  1052          1770  2 BIND
;  1053          1771  2     header = .lbr$gl_control[lbr$l_hdrptr] : BBLOCK,
;  1054          1772  2     helpinfo = .helpdata [hlp$k_info] : BBLOCK,
;  1055          1773  2     reclen = recdesc [dsc$w_length] : WORD,
;  1056          1774  2     recaddr = recdesc [dsc$a_pointer] : REF VECTOR [,BYTE];
;  1057          1775  2
;  1058          1776  2 IF .header[lhd$l_dcxmapvbn] NEQ 0
;  1059          1777  2 THEN
;  1060          1778  2     expand_record = true
;  1061          1779  2 ELSE
;  1062          1780  2     expand_record = false;
;  1063          1781  2
;  1064          1782  2 perform (print_keys (.helpdata));
;  1065          1783  2 perform (print_blankline (.helpdata));
;  1066          1784  2 CH$MOVE (rfa$c_length, helpinfo [hlp$b_readrfa], saverfa);
;  1067          1785  2 dataseen = false;
;  1068          1786  2 IF .helpinfo [hlp$l_readsts]
;  1069          1787  2 THEN
;  1070          1788  2     !
;  1071          1789  2     ! Read records until end of module or key/qualifier stop
;  1072          1790  2     !
;  1073          1791  3     WHILE  (
;  1074          1792  3             CH$MOVE (rfa$c_length, helpinfo [hlp$b_readrfa], saverfa);
;  1075          1793  4             IF (helpinfo [hlp$l_readsts] = read_record (helpinfo [hlp$b_readrfa], recdesc))
;  1076          1794  3                 AND .expand_record
;  1077          1795  3             THEN helpinfo[hlp$l_readsts] = expand_it( recdesc );
;  1078          1796  3             .helpinfo[hlp$l_readsts]
;  1079          1797  3             )
;  1080          1798  3     DO BEGIN
```

```
 4      IF (.reclen EQL 0) OR (.recaddr [0] NEQ %ASCII '!') ! We really just want to check if its a comment line
 4                  ! but we must first check if its a zero length line, because if it is, recaddr [0]
 4                  ! will be the line length of the next line instead of the first character of the current lin
 3      THEN
 4          BEGIN
 4          IF  is_key_on_line (helpinfo, recdesc, level, keydesc)
 5          THEN BEGIN
 5              IF .helpinfo [hlp$v_qualhelp]                   !If qualifier help
 6                  THEN BEGIN
 7                      IF (.helpinfo [hlp$v_qualine]               ! and its a qualifier line
 7                          AND .dataseen)                      ! and we have seen other than
 6                          OR .helpinfo [hlp$v_keyline]        !  a qualifier, or this
 6                                                              !   is a keyword line
 6                          THEN EXITLOOP;                      !     then get out of the loop
 5                      END;
 5              IF NOT .helpinfo [hlp$v_qualhelp]               !If keyword help
 5                  AND .helpinfo [hlp$v_keyline]               ! and its a keyword line
 5                  THEN EXITLOOP;                              !  then all done
 4              END;                                            !Is a key line
 4      perform (call_output (.helpdata, recdesc));
 4      helpinfo [hlp$v_anyhelp] = true;                        !Flag help was found
 4      IF NOT .helpinfo [hlp$v_qualine]                        !Unless a qualifier line
 4          THEN dataseen = true;
 3          END;                                                ! Not a comment line
 2      END;                                                    ! of while loop
 2
 2  CH$MOVE (rfa$c_length, saverfa, helpinfo [hlp$b_readrfa]); !Restore RFA of last record
 2  perform (print_options (.helpdata));                        !Print additional options available
 2
 2  RETURN true;
 1  END;                                    ! Of print_helptext
```

```
                                 0FFC 00000 PRINT_HELPTEXT:
                                                    .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11       ; 1743
                 5E          1C  C2 00002           SUBL2   #28, SP                                    ;
                 50   0000G  CF  D0 00005           MOVL    LBR$GL_CONTROL, R0                         ; 1771
                 50      0A  A0  D0 0000A           MOVL    10(R0), R0                                 ;
                 57      04  AC  D0 0000E           MOVL    HELPDATA, R7                               ; 1772
                 56      04  A7  D0 00012           MOVL    4(R7), R6                                  ;
                       008C  C0  D5 00016           TSTL    140(R0)                                    ; 1776
                             05  13 0001A           BEQL    1$                                         ;
                 59          01  D0 0001C           MOVL    #1, EXPAND_RECORD                          ; 1778
                             02  11 0001F           BRB     2$                                         ;
                             59  D4 00021 1$:       CLRL    EXPAND_RECORD                              ; 1780
                             57  DD 00023 2$:       PUSHL   R7                                         ; 1782
          0000V  CF          01  FB 00025           CALLS   #1, PRINT_KEYS                             ;
                 07          50  E9 0002A           BLBC    STATUS, 3$                                 ;
                             57  DD 0002D           PUSHL   R7                                         ; 1783
          0000V  CF          01  FB 0002F           CALLS   #1, PRINT_BLANKLINE                        ;
                 7B          50  E9 00034 3$:       BLBC    STATUS, 10$                                ;
  0C   AE     50 A6          06  28 00037           MOVC3   #6, 80(R6), SAVERFA                        ; 1784
                             58  D4 0003D           CLRL    DATASEEN                                   ; 1785
                 27      4C  A6  E9 0003F           BLBC    76(R6), 5$                                 ; 1786
```

```
          0C   AE       50    A6             06    28 00043 4$:    MOVC3   #6, 80(R6), SAVERFA                          ; 1792
                              51       14    AE    9E 00049        MOVAB   RECDESC, R1                                  ; 1793
                              50       50    A6    9E 0004D        MOVAB   80(R6), R0                                   ;
                                          0000G    30 00051        BSBW    READ_RECORD                                  ;
                        4C    A6             50    D0 00054        MOVL    R0, 76(R6)                                   ;
                              0F             50    E9 00058        BLBC    R0, 5$                                       ;
                              0C             59    E9 0005B        BLBC    EXPAND_RECORD, 5$                            ; 1794
                                       14    AE    9F 0005E        PUSHAB  RECDESC                                      ; 1795
                     0000V    CF             01    FB 00061        CALLS   #1, EXPAND_IT                                ;
                        4C    A6             50    D0 00066        MOVL    R0, 76(R6)                                   ;
                              55       4C    A6    E9 0006A 5$:    BLBC    76(R6), 11$                                  ; 1796
                                       14    AE    B5 0006E        TSTW    RECLEN                                       ; 1799
                                             06    13 00071        BEQL    6$                                           ;
                              21       18    BE    91 00073        CMPB    @RECADDR, #33                                ;
                                             CA    13 00077        BEQL    4$                                           ;
                                       04    AE    9F 00079 6$:    PUSHAB  KEYDESC                                      ; 1804
                                       04    AE    9F 0007C        PUSHAB  LEVEL                                        ;
                                       1C    AE    9F 0007F        PUSHAB  RECDESC                                      ;
                                             56    DD 00082        PUSHL   R6                                           ;
                     0000V    CF             04    FB 00084        CALLS   #4, IS_KEY_ON_LINE                           ;
                              1C             50    E9 00089        BLBC    R0, 9$                                       ;
               12       03    A6             04    E1 0008C        BBC     #4, 3(R6), 8$                                ; 1806
               03       03    A6             03    E1 00091        BBC     #3, 3(R6), 7$                                ; 1808
                              2A             58    E8 00096        BLBS    DATASEEN, 11$                                ; 1809
               25       03    A6             02    E0 00099 7$:    BBS     #2, 3(R6), 11$                               ; 1810
               05       03    A6             04    E0 0009E        BBS     #4, 3(R6), 9$                                ; 1814
               1B       03    A6             02    E0 000A3 8$:    BBS     #2, 3(R6), 11$                               ; 1815
                                       14    AE    9F 000A8 9$:    PUSHAB  RECDESC                                      ; 1818
                                             57    DD 000AB        PUSHL   R7                                           ;
                     0000V    CF             02    FB 000AD        CALLS   #2, CALL_OUTPUT                              ;
                              21             50    E9 000B2 10$:   BLBC    STATUS, 12$                                  ;
                        03    A6             01    88 000B5        BISB2   #1, 3(R6)                                    ; 1819
               85       03    A6             03    E0 000B9        BBS     #3, 3(R6), 4$                                ; 1820
                              58             01    D0 000BE        MOVL    #1, DATASEEN                                 ; 1821
                                             80    11 000C1        BRB     4$                                           ; 1791
          50   A6       0C    AE             06    28 000C3 11$:   MOVC3   #6, SAVERFA, 80(R6)                          ; 1825
                                             57    DD 000C9        PUSHL   R7                                           ; 1826
                     0000V    CF             01    FB 000CB        CALLS   #1, PRINT_OPTIONS                            ;
                              03             50    E9 000D0        BLBC    STATUS, 12$                                  ;
                              50             01    D0 000D3        MOVL    #1, R0                                       ; 1828
                                                   04 000D6 12$:   RET                                                  ; 1829

; Routine Size:  215 bytes,    Routine Base:  $CODE$ + 08AE
```

```
;  1113          1830  1 %SBTTL  'Routine print_nohelp';
;  1114          1831  1 ROUTINE print_nohelp (helpdata) =
;  1115          1832  2 BEGIN
;  1116          1833  2 !++
;  1117          1834  2 ! Tell that no help was found as requested
;  1118          1835  2 !
;  1119          1836  2 ! Inputs:
;  1120          1837  2 !
;  1121          1838  2 !       helpdata        Address of help data vector set up by lbr$get_help
;  1122          1839  2 !
;  1123          1840  2 ! Outputs:
;  1124          1841  2 !
;  1125          1842  2 !       A string telling that no help was found is output.
;  1126          1843  2 !
;  1127          1844  2 !--
;  1128          1845  2
;  1129          1846  2 MAP
;  1130          1847  2     helpdata : REF VECTOR [,LONG];
;  1131          1848  2
;  1132          1849  2 BIND
;  1133          1850  2     helpinfo = .helpdata [hlp$k_info] : BBLOCK,
;  1134          1851  2     wildflag = helpinfo [hlp$t_wildflags] : BITVECTOR;
;  1135          1852  2
;  1136          1853  2
;  1137          1854  2 LOCAL
;  1138          1855  2     lastlevel,
;  1139          1856  2     desc : BBLOCK [dsc$c_s_bln];
;  1140          1857  2
;  1141          1858  2 helpinfo [hlp$v_unohlp] = true;
;  1142          1859  2 perform (print_keys (.helpdata));
;  1143          1860  2 helpinfo [hlp$v_qualhelp] = false;
;  1144          1861  2 CH$FILL (%ASCII' ', hlp$c_maxrecsiz, .(helpinfo [hlp$l_bufdesc] + 4));
;  1145          1862  2 desc [dsc$w_length] = .nodocmsg [0];
;  1146          1863  2 desc [dsc$a_pointer] = nodocmsg [1];
;  1147          1864  2 perform (move_key (.helpdata, desc, 1));
;  1148          1865  2 lastlevel = .helpinfo [hlp$l_lastlevel];
;  1149          1866  2 IF .lastlevel EQL 0
;  1150          1867  2     THEN lastlevel = 1;
;  1151          1868  2 !
;  1152          1869  2 ! Copy as many of the keys into the buffer as we can
;  1153          1870  2 !
;  1154          1871  2 INCRU i FROM hlp$k_key1desc TO hlp$k_key1desc + .helpinfo [hlp$l_realkeys]-1    ! Print all the keys
;  1155          1872  3 DO BEGIN
;  1156          1873  3     BIND
;  1157          1874  3         curkeydesc = .helpdata [.i] : BBLOCK;
;  1158          1875  3
;  1159          1876  3     perform (move_key (.helpdata, curkeydesc, 1));
;  1160          1877  2     END;
;  1161          1878  2
;  1162          1879  2 perform (print_blankline (.helpdata));
;  1163          1880  2 perform (print_line (.helpdata));
;  1164          1881  2 perform (print_options (.helpdata));
;  1165          1882  2 helpinfo [hlp$v_unohlp] = false;
;  1166          1883  2
;  1167          1884  2 RETURN true
;  1168          1885  1 END;                                                    !Of print_nohelp
```

```
                                         01FC 00000 PRINT_NOHELP:
                                                          .WORD    Save R2,R3,R4,R5,R6,R7,R8           ; 1831
                      58     FA7C    CF   9E 00002        MOVAB    MOVE_KEY, R8                        ;
                      5E             08   C2 00007        SUBL2    #8, SP                              ;
                      57       04    AC   D0 0000A        MOVL     HELPDATA, R7                        ; 1850
                      56       04    A7   D0 0000E        MOVL     4(R7), R6                           ;
                      66             01   88 00012        BISB2    #1, (R6)                            ; 1858
                                     57   DD 00015        PUSHL    R7                                  ; 1859
                0000V CF             01   FB 00017        CALLS    #1, PRINT_KEYS                      ;
                      70             50   E9 0001C        BLBC     STATUS, 4$                          ;
                   03 A6             10   8A 0001F        BICB2    #16, 3(R6)                          ; 1860
  0050  8F       20   6E             00   2C 00023        MOVC5    #0, (SP), #32, #80, @8(R6)          ; 1861
                               08    B6      0002A                                                     ;
                      6E     F64B    CF   9B 0002C        MOVZBW   NODOCMSG, DESC                      ; 1862
                   04 AE     F647    CF   9E 00031        MOVAB    NODOCMSG+1, DESC+4                  ; 1863
                                     01   DD 00037        PUSHL    #1                                  ; 1864
                               04    AE   9F 00039        PUSHAB   DESC                                ;
                                     57   DD 0003C        PUSHL    R7                                  ;
                      68             03   FB 0003E        CALLS    #3, MOVE_KEY                        ;
                      4B             50   E9 00041        BLBC     STATUS, 4$                          ;
                      50       18    A6   D0 00044        MOVL     24(R6), LASTLEVEL                   ; 1865
                                     03   12 00048        BNEQ     1$                                  ; 1866
                      50             01   D0 0004A        MOVL     #1, LASTLEVEL                       ; 1867
              53   28 A6             04   C1 0004D 1$:    ADDL3    #4, 40(R6), R3                      ; 1871
                      52             05   D0 00052        MOVL     #5, I                               ;
                                     0F   11 00055        BRB      3$                                  ;
                                     01   DD 00057 2$:    PUSHL    #1                                  ; 1876
                                   6742   DD 00059        PUSHL    (R7)[I]                             ;
                                     57   DD 0005C        PUSHL    R7                                  ;
                      68             03   FB 0005E        CALLS    #3, MOVE_KEY                        ;
                      2B             50   E9 00061        BLBC     STATUS, 4$                          ;
                                     52   D6 00064        INCL     I                                   ; 1871
                      53             52   D1 00066 3$:    CMPL     I, R3                               ;
                                     EC   1B 00069        BLEQU    2$                                  ;
                                     57   DD 0006B        PUSHL    R7                                  ; 1879
                0000V CF             01   FB 0006D        CALLS    #1, PRINT_BLANKLINE                 ;
                      1A             50   E9 00072        BLBC     STATUS, 4$                          ;
                                     57   DD 00075        PUSHL    R7                                  ; 1880
                0000V CF             01   FB 00077        CALLS    #1, PRINT_LINE                      ;
                      10             50   E9 0007C        BLBC     STATUS, 4$                          ;
                                     57   DD 0007F        PUSHL    R7                                  ; 1881
                0000V CF             01   FB 00081        CALLS    #1, PRINT_OPTIONS                   ;
                      06             50   E9 00086        BLBC     STATUS, 4$                          ;
                      66             01   8A 00089        BICB2    #1, (R6)                            ; 1882
                      50             01   D0 0008C        MOVL     #1, R0                              ; 1884
                                          04 0008F 4$:    RET                                          ; 1885

; Routine Size:  144 bytes,    Routine Base:  $CODE$ + 0985
```

```
;  1170          1886  1 %SBTTL  'Routine print_options';
;  1171          1887  1 ROUTINE print_options (helpdata) =
;  1172          1888  2 BEGIN
;  1173          1889  2 !++
;  1174          1890  2 ! Print help options available
;  1175          1891  2 !
;  1176          1892  2 ! Inputs:
;  1177          1893  2 !
;  1178          1894  2 !       helpdata        Address of help data vector set up by lbr$get_help
;  1179          1895  2 !
;  1180          1896  2 ! Outputs:
;  1181          1897  2 !
;  1182          1898  2 !       Help that is available is output.
;  1183          1899  2 !
;  1184          1900  2 !--
```

```
; 1186          1901  2 %SBTTL  'Routine print_otherinfo';
; 1187          1902  2 ROUTINE print_otherinfo (helpdata) =
; 1188          1903  3 BEGIN
; 1189          1904  3 !++
; 1190          1905  3 ! Print the text "other information available" surrounded by
; 1191          1906  3 ! blank lines.
; 1192          1907  3 !
; 1193          1908  3 ! Inputs:
; 1194          1909  3 !
; 1195          1910  3 !       helpdata        data vector set up by lbr$get_help
; 1196          1911  3 !
; 1197          1912  3 !--
; 1198          1913  3 MAP
; 1199          1914  3     helpdata : REF VECTOR [,LONG];
; 1200          1915  3
; 1201          1916  3 LOCAL
; 1202          1917  3     desc : BBLOCK [dsc$c_s_bln];
; 1203          1918  3
; 1204          1919  3 desc [dsc$w_length] = .otherinfo [0];             !Set up descriptor for text
; 1205          1920  3 desc [dsc$a_pointer] = otherinfo [1];
; 1206          1921  3 perform (print_blankline (.helpdata));           !Print a blank line
; 1207          1922  3 perform (call_output (.helpdata, desc));         !Tell other info available
; 1208          1923  3 perform (print_blankline (.helpdata));           !and a blank line
; 1209          1924  3
; 1210          1925  3 RETURN true
; 1211          1926  2 END;                                             !Of print_otherinfo


                                        0000 00000 PRINT_OTHERINFO:
                                                          .WORD   Save nothing                         ; 1902
          5E                    0B  C2 00002              SUBL2   #8, SP                               ;
          6E        F5FE        CF  9B 00005              MOVZBW  OTHERINFO, DESC                      ; 1919
     04   AE        F5FA        CF  9E 0000A              MOVAB   OTHERINFO+1, DESC+4                  ; 1920
                      04        AC  DD 00010              PUSHL   HELPDATA                             ; 1921
  0000V   CF                    01  FB 00013              CALLS   #1, PRINT_BLANKLINE                  ;
          1B                    50  E9 00018              BLBC    STATUS, 1$                           ;
                                5E  DD 0001B              PUSHL   SP                                   ; 1922
                      04        AC  DD 0001D              PUSHL   HELPDATA                             ;
  0000V   CF                    02  FB 00020              CALLS   #2, CALL_OUTPUT                      ;
          0E                    50  E9 00025              BLBC    STATUS, 1$                           ;
                      04        AC  DD 00028              PUSHL   HELPDATA                             ; 1923
  0000V   CF                    01  FB 0002B              CALLS   #1, PRINT_BLANKLINE                  ;
          03                    50  E9 00030              BLBC    STATUS, 1$                           ;
          50                    01  D0 00033              MOVL    #1, R0                               ; 1925
                                    04 00036  1$:         RET                                          ; 1926

; Routine Size:  55 bytes,    Routine Base:  $CODE$ + 0A15
```

```
2 %SBTTL  'Routine move_watch_tabs';
2 ROUTINE move_watch_tabs (helpdata, desc) =
3 BEGIN
3 !++
3 !
3 ! Move a key into the buffer with logical tab control
3 !
3 ! Inputs:
3 !
3 !       helpdata        Address of help data vector set up by lbr$get_help
3 !       desc            Address of string descriptor for key
3 !
3 ! Outputs:
3 !
3 !       Key is copied into the buffer, watching logical tab stops
3 !
3 !--
3
3 MAP
3     helpdata : REF VECTOR [,LONG],
3     desc : REF BBLOCK;
3
3 BIND
3     helpinfo = .helpdata [hlp$k_info] : BBLOCK;
3
3 LOCAL
3     endpos,
3     startpos,
3     keytabs;
3
3 startpos = .helpinfo  [hlp$l_tabindex] * hlp$c_logtab;
3 endpos = .helpinfo [hlp$l_width] - ((.helpinfo [hlp$l_curlevel] + 1) * hlp$c_indent);
3 IF .startpos GEQU .endpos
3     OR (.startpos + .desc [dsc$w_length] + 1) GTRU .endpos
3 THEN  perform (print_line (.helpdata));          !Make room for line
3 keytabs = (.desc [dsc$w_length] + hlp$c_logtab) / hlp$c_logtab;
3 helpinfo [hlp$l_tabindex] = .helpinfo [hlp$l_tabindex] + .keytabs;
3 helpinfo [hlp$l_curptr] = CH$COPY (.desc [dsc$w_length],
3                           .desc [dsc$a_pointer], %ASCII ' ',
3                           .keytabs*hlp$c_logtab, .helpinfo [hlp$l_curptr]);
3 helpinfo [hlp$l_nchars] = .helpinfo [hlp$l_nchars] + .keytabs*hlp$c_logtab;
3 RETURN true;
2 END;                                    !Of move_watch_tabs
```

```
                          00FC 00000 MOVE_WATCH_TABS:
                                         .WORD   Save R2,R3,R4,R5,R6,R7       ; 1928
            53       04  AC  D0 00002    MOVL    HELPDATA, R3                 ; 1950
            56       04  A3  D0 00006    MOVL    4(R3), R6
   52   1C  A6           0B  C5 0000A    MULL3   #11, 28(R6), STARTPOS        ; 1957
            50       14  A6  D0 0000F    MOVL    20(R6), R0                   ; 1958
            50           02  C4 00013    MULL2   #2, R0
   50   20  A6           50  C3 00016    SUBL3   R0, 32(R6), R0
            50           02  C2 0001B    SUBL2   #2, ENDPOS                   ;
```

```
                     50           52  D1 0001E         CMPL    STARTPOS, ENDPOS                       ; 1959
                                  0E  1E 00021         BGEQU   1$                                     ;
                     51      08   BC  3C 00023         MOVZWL  @DESC, R1                              ; 1960
                     51      01 A142  9E 00027         MOVAB   1(R1)[STARTPOS], R1                    ;
                     50           51  D1 0002C         CMPL    R1, ENDPOS                             ;
                                  0A  1B 0002F         BLEQU   2$                                     ;
                                  53  DD 00031  1$:    PUSHL   R3                                     ; 1961
                 0000V   CF       01  FB 00033         CALLS   #1, PRINT_LINE                         ;
                     28           50  E9 00038         BLBC    STATUS, 3$                             ;
                     51      08   AC  D0 0003B  2$:    MOVL    DESC, R1                               ; 1962
                     50           61  3C 0003F         MOVZWL  (R1), R0                               ;
                     50           0B  C0 00042         ADDL2   #11, R0                                ;
                     50           0B  C6 00045         DIVL2   #11, KEYTABS                           ;
                1C   A6           50  C0 00048         ADDL2   KEYTABS, 28(R6)                        ; 1963
           57        50           0B  C5 0004C         MULL3   #11, KEYTABS, R7                       ; 1966
   57      20   04   B1           61  2C 00050         MOVC5   (R1), @4(R1), #32, R7, @12(R6)         ;
                             0C   B6     00056                                                        ;
                0C   A6           53  D0 00058         MOVL    R3, 12(R6)                             ;
                10   A6           57  C0 0005C         ADDL2   R7, 16(R6)                             ; 1967
                     50           01  D0 00060         MOVL    #1, R0                                 ; 1968
                                  04     00063  3$:    RET                                            ; 1969
```

; Routine Size: 100 bytes,   Routine Base: $CODE$ + 0A4C

```
; 1257        1970  2 %SBTTL  'Routine add_key';
; 1258        1971  2 ROUTINE add_key  (entry, user_routine, index_desc, helpdata) =
; 1259        1972  3 BEGIN
; 1260        1973  3 !
; 1261        1974  3 ! Move a key into the buffer
; 1262        1975  3 !
; 1263        1976  3 MAP
; 1264        1977  3     entry : REF BBLOCK,
; 1265        1978  3     helpdata : REF VECTOR [,LONG];
; 1266        1979  3
; 1267        1980  3 LOCAL
; 1268        1981  3     entrydesc : BBLOCK [dsc$c_s_bln];
; 1269        1982  3
; 1270        1983  3 entrydesc [dsc$w_length] = .entry [idx$b_keylen];
; 1271        1984  3 entrydesc [dsc$a_pointer] = entry [idx$t_keyname];
; 1272        1985  3 perform (move_watch_tabs (.helpdata, entrydesc));
; 1273        1986  3 RETURN true
; 1274        1987  2 END;                                    !Of add_key



                                  0000 00000 ADD_KEY:.WORD    Save nothing                    ; 1971
            5E                 08   C2 00002        SUBL2    #8, SP                          ;
            50         04      AC   D0 00005        MOVL     ENTRY, R0                       ; 1983
            6E         06      A0   9B 00009        MOVZBW   6(R0), ENTRYDESC                ;
     04     AE         07      A0   9E 0000D        MOVAB    7(R0), ENTRYDESC+4              ; 1984
                               5E   DD 00012        PUSHL    SP                              ; 1985
                       10      AC   DD 00014        PUSHL    HELPDATA                        ;
     81     AF                 02   FB 00017        CALLS    #2, MOVE_WATCH_TABS             ;
            03                 50   E9 0001B        BLBC     STATUS, 1$                      ;
            50                 01   D0 0001E        MOVL     #1, R0                          ; 1986
                                    04 00021 1$:    RET                                      ; 1987

; Routine Size:  34 bytes,    Routine Base:  $CODE$ + 0AB0
```

```
;  1276     1988  2 %SBTTL  'Main body of print_options';
;  1277     1989  2
;  1278     1990  2 !
;  1279     1991  2 ! Main body of print_options
;  1280     1992  2 !
;  1281     1993  2 MAP
;  1282     1994  2     helpdata : REF VECTOR [,LONG];
;  1283     1995  2
;  1284     1996  2 LOCAL
;  1285     1997  2     expand_record,
;  1286     1998  2     lastflags,
;  1287     1999  2     level,
;  1288     2000  2     lastlevel,
;  1289     2001  2     tokendesc : BBLOCK [dsc$c_s_bln],
;  1290     2002  2     recdesc : BBLOCK [dsc$c_s_bln],
;  1291     2003  2     desc : BBLOCK [dsc$c_s_bln],
;  1292     2004  2     saverfa : BBLOCK [rfa$c_length],
;  1293     2005  2     first_time;
;  1294     2006  2
;  1295     2007  2 BIND
;  1296     2008  2     header = .lbr$gl_control[lbr$l_hdrptr] : BBLOCK,
;  1297     2009  2     helpinfo = .helpdata [hlp$k_info] : BBLOCK,
;  1298     2010  2     curflags = helpinfo [hlp$l_hlpflags] + 2 : WORD,
;  1299     2011  2     key2rfa = helpinfo [hlp$b_key2rfa],
;  1300     2012  2     wildflag = helpinfo [hlp$t_wildflags] : BITVECTOR,
;  1301     2013  2     reclen = recdesc [dsc$w_length] : WORD,
;  1302     2014  2     recaddr = recdesc [dsc$a_pointer];
;  1303     2015  2
;  1304     2016  2 IF .header[lhd$l_dcxmapvbn] NEQ 0
;  1305     2017  2 THEN
;  1306     2018  2     expand_record = true
;  1307     2019  2 ELSE
;  1308     2020  2     expand_record = false;
;  1309     2021  2
;  1310     2022  2 IF .helpinfo [hlp$v_qualhelp] OR .helpinfo [hlp$v_allhelp]
;  1311     2023  2     THEN RETURN true;
;  1312     2024  2
;  1313     2025  2 lastlevel = .helpinfo [hlp$l_lastlevel];
;  1314     2026  2 IF .helpinfo [hlp$v_unohlp]
;  1315     2027  2     AND .lastlevel NEQ 0
;  1316     2028  2         THEN DO lastlevel = .lastlevel - 1
;  1317     2029  4                 UNTIL ((.lastlevel EQL 0)
;  1318     2030  2                                 OR NOT .wildflag [.lastlevel - 1]);
;  1319     2031  2
;  1320     2032  2 helpinfo [hlp$v_uothinfo] = true;
;  1321     2033  2 IF .lastlevel EQL 1
;  1322     2034  2     AND .helpinfo [hlp$v_unohlp]
;  1323     2035  2     AND .key2rfa NEQ 0                  ! avoid storing an rfa which has never been set
;  1324     2036  2     THEN CH$MOVE (rfa$c_length, helpinfo [hlp$b_key2rfa], helpinfo [hlp$b_readrfa])
;  1325     2037  2     ELSE CH$MOVE (rfa$c_length, helpinfo [hlp$b_lstkeyrfa], helpinfo [hlp$b_readrfa]);
;  1326     2038  2 first_time = true;
;  1327     2039  2 lastflags = 0;
;  1328     2040  2 level = 0;
;  1329     2041  2
;  1330     2042  3 IF (.helpinfo [hlp$v_unohlp]                        !If first no help found
;  1331     2043  3     AND .lastlevel EQL 0)                          ! at first level
;  1332     2044  2     OR .helpinfo [hlp$v_helphlp]                   ! or inserted 'HELP' key
```

```
;  1333     2045  3 THEN BEGIN
;  1334     2046  3     helpinfo [hlp$v_anyhelp] = true;
;  1335     2047  3     perform (print_otherinfo (.helpdata));                  !Print "other info available"
;  1336     2048  3     perform (traverse_keys (1, add_key, 0, .helpdata ); !Format and print index
;  1337     2049  3     IF .helpinfo [hlp$l_nchars] NEQ 0
;  1338     2050  3         THEN perform (print_line (.helpdata));
;  1339     2051  3     END
;  1340     2052  3
;  1341     2053  3 ELSE
;  1342     2054  3 WHILE  (
;  1343     2055  4          CH$MOVE (rfa$c_length, helpinfo [hlp$b_readrfa], saverfa);
;  1344     2056  4          IF (helpinfo [hlp$l_readsts] = read_record (helpinfo [hlp$b_readrfa], recdesc))
;  1345     2057  4              AND .expand_record
;  1346     2058  4          THEN helpinfo[hlp$l_readsts] = expand_it( recdesc );
;  1347     2059  4          .helpinfo[hlp$l_readsts]
;  1348     2060  4          )
;  1349     2061  3 DO IF is_key_on_line (helpinfo, recdesc, level, tokendesc)      !If key on line
;  1350     2062  3     AND (IF .helpinfo [hlp$v_qualhelp]                          ! (if qualifier help
;  1351     2063  3                 THEN .helpinfo [hlp$v_qualine]                 !   and its a qualifier
;  1352     2064  3                 ELSE (
;  1353     2065  4                         IF .first_time AND .helpinfo [hlp$v_qualine]
;  1354     2066  4                             THEN false
;  1355     2067  4                             ELSE true
;  1356     2068  4                         )
;  1357     2069  4                         )
;  1358     2070  3     AND .level LEQ .lastlevel + 1                               !And we might want to look at key
;  1359     2071  3 THEN BEGIN
;  1360     2072  3     IF .level LEQ .lastlevel                                    !If found start of next level
;  1361     2073  3     THEN BEGIN
;  1362     2074  4         CH$MOVE (rfa$c_length, saverfa, helpinfo [hlp$b_readrfa]);      !Restore RFA of last record
;  1363     2075  4         IF .helpinfo [hlp$l_nchars] NEQ 0
;  1364     2076  4         THEN perform (print_line (.helpdata));
;  1365     2077  4         RETURN true;
;  1366     2078  3         END;
;  1367     2079  3     IF .first_time
;  1368     2080  4     THEN BEGIN
;  1369     2081  4         perform (print_otherinfo (.helpdata));  !Print "other info available"
;  1370     2082  4         helpinfo [hlp$v_anyhelp] = true;        !Flag help was found
;  1371     2083  4         first_time = false;
;  1372     2084  3         END;
;  1373     2085  5     IF ((.lastflags NEQ .curflags)                              !If different line type
;  1374     2086  4         AND (.lastflags NEQ 0))                                 ! (and not first line)
;  1375     2087  3         THEN perform (print_line (.helpdata));                  ! then force out previous line
;  1376     2088  3     tokendesc [dsc$w_length] = .reclen -                        !Figure length of line
;  1377     2089  3                                (.tokendesc [dsc$a_pointer] - .recaddr);
;  1378     2090  3     perform (move_watch_tabs (.helpdata, tokendesc));
;  1379     2091  3     lastflags = .curflags;                                      !Set new flags
;  1380     2092  2     END;
;  1381     2093  2
;  1382     2094  2 CH$MOVE (rfa$c_length, saverfa, helpinfo [hlp$b_readrfa]);
;  1383     2095  2 IF .helpinfo [hlp$l_nchars] NEQ 0
;  1384     2096  2     THEN perform (print_line (.helpdata));
;  1385     2097  2
;  1386     2098  2 helpinfo[hlp$v_uothinfo] = false;       ! Reset otherinfo flag
;  1387     2099  2
;  1388     2100  2 RETURN true
;  1389     2101  1 END;                                                           !Of print_options
```

```
                                  0FFC 00000 PRINT_OPTIONS:
                                                         .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11      ; 1887
                  5E          2C   C2 00002              SUBL2   #44, SP
                  50   0000G  CF   D0 00005              MOVL    LBR$GL_CONTROL, R0                        ; 2008
                  50      0A  A0   D0 0000A              MOVL    10(R0), R0
                  58      04  AC   D0 0000E              MOVL    HELPDATA, R8                              ; 2009
                  57      04  A8   D0 00012              MOVL    4(R8), R7
                  59      02  A7   9E 00016              MOVAB   2(R7), R9                                 ; 2010
                        008C  C0   D5 0001A              TSTL    140(R0)                                   ; 2016
                              05   13 0001E              BEQL    1$
                  6E          01   D0 00020              MOVL    #1, EXPAND_RECORD                         ; 2018
                              02   11 00023              BRB     2$
                              6E   D4 00025 1$:          CLRL    EXPAND_RECORD                             ; 2020
   03             69          0C   E1 00027 2$:          BBC     #12, (R9), 4$                             ; 2022
                            0156   31 0002B 3$:          BRW     24$
   F9             69          0E   E0 0002E 4$:          BBS     #14, (R9), 3$
                  56      18  A7   D0 00032              MOVL    24(R7), LASTLEVEL                         ; 2025
                  0F          67   E9 00036              BLBC    (R7), 6$                                  ; 2026
                              0D   13 00039              BEQL    6$                                        ; 2027
                              56   D7 0003B 5$:          DECL    LASTLEVEL                                 ; 2028
                              09   13 0003D              BEQL    6$                                        ; 2029
                  50      FF  A6   9E 0003F              MOVAB   -1(R6), R0                                ; 2030
   F3        44   A7          50   E0 00043              BBS     R0, 68(R7), 5$
                  67          04   88 00048 6$:          BISB2   #4, (R7)                                  ; 2032
                  01          56   D1 0004B              CMPL    LASTLEVEL, #1                             ; 2033
                              13   12 0004E              BNEQ    7$
                  10          67   E9 00050              BLBC    (R7), 7$                                  ; 2034
                          3E  A7   D5 00053              TSTL    62(R7)                                    ; 2035
                              0B   13 00056              BEQL    7$
                  5A      50  A7   9E 00058              MOVAB   80(R7), R10                               ; 2036
   6A        3E   A7          06   28 0005C              MOVC3   #6, 62(R7), (R10)
                              09   11 00061              BRB     8$
                  5A      50  A7   9E 00063 7$:          MOVAB   80(R7), R10                               ; 2037
   6A        56   A7          06   28 00067              MOVC3   #6, 86(R7), (R10)
                  5B          01   D0 0006C 8$:          MOVL    #1, FIRST_TIME                            ; 2038
                          04  AE   7C 0006F              CLRQ    LASTFLAGS                                 ; 2039
                  04          67   E9 00072              BLBC    (R7), 9$                                  ; 2042
                              56   D5 00075              TSTL    LASTLEVEL                                 ; 2043
                              04   13 00077              BEQL    10$
   34             69          09   E1 00079 9$:          BBC     #9, (R9), 15$                             ; 2044
             01   A9          01   88 0007D 10$:         BISB2   #1, 1(R9)                                 ; 2046
                              58   DD 00081              PUSHL   R8                                        ; 2047
           FEBB   CF          01   FB 00083              CALLS   #1, PRINT_OTHERINFO
                  0F          50   E9 00088              BLBC    STATUS, 11$
                              58   DD 0008B              PUSHL   R8                                        ; 2048
                              7E   D4 0008D              CLRL    -(SP)
                        FF4B  CF   9F 0008F              PUSHAB  ADD_KEY
                              01   DD 00093              PUSHL   #1
          0000G   CF          04   FB 00095              CALLS   #4, TRAVERSE_KEYS
                  01          50   E8 0009A 11$:         BLBS    STATUS, 12$
                                   04 0009D              RET
                          10  A7   D5 0009E 12$:         TSTL    16(R7)                                    ; 2049
```

```
                                    03  12 000A1        BNEQ    14$                                  ;
                                  00C7  31 000A3 13$:   BRW     22$                                  ;
                                    58  DD 000A6 14$:   PUSHL   R8                                   ; 2050
                 0000V  CF          01  FB 000A8        CALLS   #1, PRINT_LINE                       ;
                        F3          50  E8 000AD        BLBS    STATUS, 13$                          ;
                                        04 000B0        RET                                          ;
        0C  AE          6A          06  28 000B1 15$:   MOVC3   #6, (R10), SAVERFA                   ; 2055
                        51      1C  AE  9E 000B6        MOVAB   RECDESC, R1                          ; 2056
                        50          5A  D0 000BA        MOVL    R10, R0                              ;
                                 0000G  30 000BD        BSBW    READ_RECORD                          ;
                   4C   A7          50  D0 000C0        MOVL    R0, 76(R7)                           ;
                        0F          50  E9 000C4        BLBC    R0, 16$                              ;
                        0C          6E  E9 000C7        BLBC    EXPAND_RECORD, 16$                   ; 2057
                                1C  AE  9F 000CA        PUSHAB  RECDESC                              ; 2058
                 0000V  CF          01  FB 000CD        CALLS   #1, EXPAND_IT                        ;
                   4C   A7          50  D0 000D2        MOVL    R0, 76(R7)                           ;
                        C9      4C  A7  E9 000D6 16$:   BLBC    76(R7), 13$                          ; 2059
                                24  AE  9F 000DA        PUSHAB  TOKENDESC                            ; 2061
                                0C  AE  9F 000DD        PUSHAB  LEVEL                                ;
                                24  AE  9F 000E0        PUSHAB  RECDESC                              ;
                                    57  DD 000E3        PUSHL   R7                                   ;
                 0000V  CF          04  FB 000E5        CALLS   #4, IS_KEY_ON_LINE                   ;
                        C4          50  E9 000EA        BLBC    R0, 15$                              ;
            06          69          0C  E1 000ED        BBC     #12, (R9), 17$                       ; 2062
            BC          69          0B  E1 000F1        BBC     #11, (R9), 15$                       ; 2063
                                    07  11 000F5        BRB     18$                                  ;
                        04          5B  E9 000F7 17$:   BLBC    FIRST_TIME, 18$                      ; 2065
            B3          69          0B  E0 000FA        BBS     #11, (R9), 15$                       ;
                        50      01  A6  9E 000FE 18$:   MOVAB   1(R6), R0                            ; 2070
                        50      08  AE  D1 00102        CMPL    LEVEL, R0                            ;
                                    A9  14 00106        BGTR    15$                                  ;
                        56      08  AE  D1 00108        CMPL    LEVEL, LASTLEVEL                     ; 2072
                                    15  14 0010C        BGTR    19$                                  ;
            6A     0C   AE          06  28 0010E        MOVC3   #6, SAVERFA, (R10)                   ; 2074
                                10  A7  D5 00113        TSTL    16(R7)                               ; 2075
                                    6C  13 00116        BEQL    24$                                  ;
                                    58  DD 00118        PUSHL   R8                                   ; 2076
                 0000V  CF          01  FB 0011A        CALLS   #1, PRINT_LINE                       ;
                        62          50  E8 0011F        BLBS    STATUS, 24$                          ;
                                        04 00122        RET                                          ; 2077
                        10          5B  E9 00123 19$:   BLBC    FIRST_TIME, 20$                      ; 2079
                                    58  DD 00126        PUSHL   R8                                   ; 2081
                  FE16  CF          01  FB 00128        CALLS   #1, PRINT_OTHERINFO                  ;
                        57          50  E9 0012D        BLBC    STATUS, 25$                          ;
                   01   A9          01  88 00130        BISB2   #1, 1(R9)                            ; 2082
                                    5B  D4 00134        CLRL    FIRST_TIME                           ; 2083
04  AE      69          10          00  ED 00136 20$:   CMPZV   #0, #16, (R9), LASTFLAGS             ; 2085
                                    0F  13 0013C        BEQL    21$                                  ;
                                04  AE  D5 0013E        TSTL    LASTFLAGS                            ; 2086
                                    0A  13 00141        BEQL    21$                                  ;
                                    58  DD 00143        PUSHL   R8                                   ; 2087
                 0000V  CF          01  FB 00145        CALLS   #1, PRINT_LINE                       ;
                        3A          50  E9 0014A        BLBC    STATUS, 25$                          ;
            50     20   AE      28  AE  C3 0014D 21$:   SUBL3   TOKENDESC+4, RECADDR, R0             ; 2089
        24  AE          50      1C  AE  A1 00153        ADDW3   RECLEN, R0, TOKENDESC                ;
                                24  AE  9F 00159        PUSHAB  TOKENDESC                            ; 2090
                                    58  DD 0015C        PUSHL   R8                                   ;
```

```
                         FE17   CF              02  FB 0015E          CALLS   #2, MOVE_WATCH_TABS                          ;
                                21              50  E9 00163          BLBC    STATUS, 25$                                 ;
                           04   AE              69  3C 00166          MOVZWL  (R9), LASTFLAGS                             ; 2091
                                              FF44  31 0016A          BRW     15$                                         ; 2061
                     6A    0C   AE              06  28 0016D 22$:     MOVC3   #6, SAVERFA, (R10)                          ; 2094
                                          10  A7  D5 00172          TSTL    16(R7)                                      ; 2095
                                              0A  13 00175          BEQL    23$                                         ;
                                              58  DD 00177          PUSHL   R8                                          ; 2096
                         0000V  CF              01  FB 00179          CALLS   #1, PRINT_LINE                              ;
                                06              50  E9 0017E          BLBC    STATUS, 25$                                 ;
                                67              04  8A 00181 23$:     BICB2   #4, (R7)                                    ; 2098
                                50              01  D0 00184 24$:     MOVL    #1, R0                                      ; 2100
                                                    04 00187 25$:     RET                                                 ; 2101

; Routine Size:  392 bytes,    Routine Base:  $CODE$ + 0AD2
```

```
;  1391          2102  1 %SBTTL  'Routine print_keys';
;  1392          2103  1 ROUTINE print_keys (helpdata) =
;  1393          2104  2 BEGIN
;  1394          2105  2 !++
;  1395          2106  2 ! Print the keys found
;  1396          2107  2 !
;  1397          2108  2 ! Inputs:
;  1398          2109  2 !
;  1399          2110  2 !       helpdata        Address of help data vector set up by lbr$get_help
;  1400          2111  2 !
;  1401          2112  2 ! Implicit inputs:
;  1402          2113  2 !
;  1403          2114  2 !       The keylist array is set up.
;  1404          2115  2 !
;  1405          2116  2 ! Outputs:
;  1406          2117  2 !
;  1407          2118  2 !       The key names are displayed on the terminal
;  1408          2119  2 !
;  1409          2120  2 !--
;  1410          2121  2
;  1411          2122  2 MAP
;  1412          2123  2     helpdata : REF VECTOR [,LONG];
;  1413          2124  2
;  1414          2125  2 LOCAL
;  1415          2126  2     lastlevel;
;  1416          2127  2
;  1417          2128  2 BIND
;  1418          2129  2     helpinfo = .helpdata [hlp$k_info] : BBLOCK,
;  1419          2130  2     wildflag = helpinfo [hlp$t_wildflags] : BITVECTOR,
;  1420          2131  2     keylist = .helpinfo [hlp$l_keylist] : BBLOCK;
;  1421          2132  2
;  1422          2133  2 IF (lastlevel = .helpinfo [hlp$l_lastlevel]) EQL 0       !If no keys found
;  1423          2134  2     THEN RETURN true;                                    ! then don't print any
;  1424          2135  2 helpinfo [hlp$v_ukeylin] = true;                        !Flag on keyname line
;  1425          2136  2
;  1426          2137  2 IF .helpinfo [hlp$v_unohlp]                            !If no help found
;  1427          2138  2     THEN DO lastlevel = .lastlevel - 1
;  1428          2139  4         UNTIL ((.lastlevel EQL 0)
;  1429          2140  2             OR NOT .wildflag [.lastlevel - 1]);
;  1430          2141  2
;  1431          2142  2 lastlevel = .lastlevel - 1;                            !Adjust for 0 base
;  1432          2143  2 IF .lastlevel GEQ 0
;  1433          2144  2 THEN INCR i FROM 0 TO .lastlevel                       !Loop through all descriptors
;  1434          2145  3 DO BEGIN
;  1435          2146  3     BIND
;  1436          2147  3         curkeydesc = keylist + .i*dsc$c_s_bln : BBLOCK; !Point to the descriptor
;  1437          2148  3
;  1438          2149  3     IF .curkeydesc [dsc$a_pointer] NEQ 0                 !If valid descriptor
;  1439          2150  4     THEN BEGIN
;  1440          2151  4         helpinfo [hlp$l_curlevel] = .i + 1;              !Set correct help level
;  1441          2152  4         perform (print_blankline (.helpdata));           !Print blank line
;  1442          2153  4         perform (call_output (.helpdata, curkeydesc));   !Print the key line
;  1443          2154  3         END;
;  1444          2155  2     END;
;  1445          2156  2
;  1446          2157  2 helpinfo [hlp$v_ukeylin] = false;                       !No longer a key line
;  1447          2158  2 RETURN true
```

```
; 1448      2159  2
; 1449      2160  1 END;                                              !of print_keys



                                        00FC 00000 PRINT_KEYS:
                                                        .WORD   Save R2,R3,R4,R5,R6,R7                    ; 2103
                     56      04   AC  D0 00002          MOVL    HELPDATA, R6                              ; 2129
                     53      04   A6  D0 00006          MOVL    4(R6), R3                                 ;
                     57      24   A3  D0 0000A          MOVL    36(R3), R7                                ; 2131
                     52      18   A3  D0 0000E          MOVL    24(R3), LASTLEVEL                         ; 2133
                                  47  13 00012          BEQL    6$                                        ;
                     63           02  88 00014          BISB2   #2, (R3)                                  ; 2135
                     0D           63  E9 00017          BLBC    (R3), 2$                                  ; 2137
                                  52  D7 0001A 1$:      DECL    LASTLEVEL                                 ; 2138
                                  09  13 0001C          BEQL    2$                                        ; 2139
                     50      FF   A2  9E 0001E          MOVAB   -1(R2), R0                                ; 2140
         F3     44   A3           50  E0 00022          BBS     R0, 68(R3), 1$                            ;
                                  52  D7 00027 2$:      DECL    LASTLEVEL                                 ; 2142
                                  2D  19 00029          BLSS    5$                                        ; 2143
                     54           01  CE 0002B          MNEGL   #1, I                                     ; 2144
                                  24  11 0002E          BRB     4$                                        ;
                     55         6744  7E 00030 3$:      MOVAQ   (R7)[I], R5                               ; 2147
                             04   A5  D5 00034          TSTL    4(R5)                                     ; 2149
                                  1B  13 00037          BEQL    4$                                        ;
                14   A3      01   A4  9E 00039          MOVAB   1(R4), 20(R3)                             ; 2151
                                  56  DD 0003E          PUSHL   R6                                        ; 2152
             0000V   CF           01  FB 00040          CALLS   #1, PRINT_BLANKLINE                       ;
                     16           50  E9 00045          BLBC    STATUS, 7$                                ;
                                  55  DD 00048          PUSHL   R5                                        ; 2153
                                  56  DD 0004A          PUSHL   R6                                        ;
             0000V   CF           02  FB 0004C          CALLS   #2, CALL_OUTPUT                           ;
                     0A           50  E9 00051          BLBC    STATUS, 7$                                ;
         D8          54           52  F3 00054 4$:      AOBLEQ  LASTLEVEL, I, 3$                          ; 2144
                     63           02  8A 00058 5$:      BICB2   #2, (R3)                                  ; 2157
                     50           01  D0 0005B 6$:      MOVL    #1, R0                                    ; 2158
                                      04 0005E 7$:      RET                                               ; 2160

; Routine Size:  95 bytes,    Routine Base:  $CODE$ + 0C5A
```

```
;  1451          2161  1 %SBTTL  'Routine print_line';
;  1452          2162  1 ROUTINE print_line (helpdata) =
;  1453          2163  2 BEGIN
;  1454          2164  2 !++
;  1455          2165  2 ! Print the line
;  1456          2166  2 !
;  1457          2167  2 ! Inputs:
;  1458          2168  2 !
;  1459          2169  2 !       helpdata        Address of help data vector set up by lbr$get_help
;  1460          2170  2 !
;  1461          2171  2 ! Implicit inputs:
;  1462          2172  2 !
;  1463          2173  2 !       the buffer descriptor in the helpinfo vector has a valid string descriptor
;  1464          2174  2 !
;  1465          2175  2 ! Outputs:
;  1466          2176  2 !
;  1467          2177  2 !       String is output
;  1468          2178  2 !
;  1469          2179  2 !--
;  1470          2180  2
;  1471          2181  2 MAP
;  1472          2182  2     helpdata : REF VECTOR [,LONG];
;  1473          2183  2
;  1474          2184  2 BIND
;  1475          2185  2     helpinfo = .helpdata [hlp$k_info] : BBLOCK;
;  1476          2186  2
;  1477          2187  2 LOCAL
;  1478          2188  2     desc : BBLOCK [dsc$c_s_bln];
;  1479          2189  2
;  1480          2190  2 desc [dsc$a_pointer] = .(helpinfo [hlp$l_bufdesc] + 4);
;  1481          2191  2 desc [dsc$w_length] = .helpinfo [hlp$l_curptr] - .desc [dsc$a_pointer];
;  1482          2192  2 perform (call_output (.helpdata, desc));
;  1483          2193  2 helpinfo [hlp$l_nchars] = 0;                          !Reset the counter
;  1484          2194  2 helpinfo [hlp$l_curptr] = .(helpinfo [hlp$l_bufdesc] + 4);!and pointer
;  1485          2195  2 CH$FILL (%ASCII' ', .desc [dsc$w_length], .desc [dsc$a_pointer]);
;  1486          2196  2 RETURN true
;  1487          2197  1 END;                                                  !Of print_line


                                    003C 00000 PRINT_LINE:
                                                        .WORD   Save R2,R3,R4,R5                        ; 2162
                                5E          08  C2 00002        SUBL2   #8, SP                          ;
                                50      04  AC  D0 00005        MOVL    HELPDATA, R0                    ; 2185
                                52      04  A0  D0 00009        MOVL    4(R0), R2                       ;
                          04    AE      08  A2  D0 0000D        MOVL    8(R2), DESC+4                   ; 2190
                    6E    0C    A2      04  AE  A3 00012        SUBW3   DESC+4, 12(R2), DESC            ; 2191
                                      4001  8F  BB 00018        PUSHR   #^M<R0,SP>                      ; 2192
                          0000V     CF      02  FB 0001C        CALLS   #2, CALL_OUTPUT                 ;
                                12          50  E9 00021        BLBC    STATUS, T$                      ;
                                          10  A2  D4 00024        CLRL    16(R2)                          ; 2193
                          0C    A2      08  A2  D0 00027        MOVL    8(R2), 12(R2)                   ; 2194
                20        6E          00  2C 0002C        MOVC5   #0, (SP), #32, DESC, @DESC+4    ; 2195
                                      04  BE      00031       ;
                                50          01  D0 00033        MOVL    #1, R0                          ; 2196
```

```
                                04 00036 1$:     RET                                        ; 2197

; Routine Size:  55 bytes,    Routine Base:  $CODE$ + 0CB9
```

```
2198   1 %SBTTL  'Routine print_blankline';
2199   1 ROUTINE print_blankline (helpdata) =
2200   2 BEGIN
2201   2 !++
2202   2 ! Print a blank line
2203   2 !
2204   2 ! Inputs:
2205   2 !
2206   2 !       helpdata        Address of help data vector set up by lbr$get_help
2207   2 !
2208   2 ! Outputs:
2209   2 !
2210   2 !       A blank line is output.
2211   2 !
2212   2 !--
2213   2
2214   2 MAP
2215   2     helpdata : REF VECTOR [,LONG];
2216   2
2217   2 LOCAL
2218   2     desc : BBLOCK [dsc$c_s_bln];
2219   2
2220   2 CH$FILL (0, dsc$c_s_bln, desc);
2221   2 RETURN call_output (.helpdata, desc)
2222   1 END;                                          !of print_blankline


                                          003C 00000 PRINT_BLANKLINE:
                                                            .WORD   Save R2,R3,R4,R5                      ; 2199
                            5E           08  C2 00002       SUBL2   #8, SP                                ;
     08          00         6E           00  2C 00005       MOVC5   #0, (SP), #0, #8, DESC                ; 2220
                                         6E     0000A
                                         5E  DD 0000B       PUSHL   SP                                    ; 2221
                                    04   AC  DD 0000D       PUSHL   HELPDATA                              ;
                      0000V CF           02  FB 00010       CALLS   #2, CALL_OUTPUT                       ;
                                             04 00015       RET                                           ; 2222

; Routine Size:  22 bytes,    Routine Base:  $CODE$ + 0CF0
```

```
%SBTTL  'Routine call_output';
ROUTINE call_output (helpdata, desc) =
BEGIN
!++
! Call user routine or LIB$PUT_OUTPUT to print line of help text.
!
! Inputs:
!
!       helpdata        Address of help data vector set up by lbr$get_help
!       desc            Address of string descriptor of line to output
!
! Outputs:
!
!       line is output via lib$put_output or user routine
!
!--

MAP
    helpdata : REF VECTOR [,LONG],
    desc : REF BBLOCK;

LOCAL
    flags,
    ptr,
    spaces,
    localdesc : BBLOCK [dsc$c_s_bln],
    a_zero,
    linebuffer : BBLOCK [hlp$c_maxrecsiz*2];

BIND
    helpinfo = .helpdata [hlp$k_info] : BBLOCK,
    linedesc = helpinfo [hlp$l_bufdesc] : BBLOCK,
    user_data = (
            IF .helpdata [hlp$k_userdata] NEQ 0
                THEN .helpdata [hlp$k_userdata]
                ELSE a_zero
            );

BIND ROUTINE
    typeout_routine = helpdata [hlp$k_userout];

a_zero = 0;
CH$FILL (0, dsc$c_s_bln, localdesc);
IF .desc [dsc$w_length] NEQ 0
    AND .desc [dsc$a_pointer] NEQ 0
THEN BEGIN
    IF .helpinfo [hlp$v_ukeylin] OR (.typeout_routine NEQ 0)
        THEN spaces = 0
        ELSE spaces = (.helpinfo [hlp$l_curlevel] + 1) * hlp$c_indent;
    ptr = CH$FILL (%ASCII ' ', .spaces, linebuffer);
    CH$MOVE (.desc [dsc$w_length], .desc [dsc$a_pointer], .ptr);
    localdesc [dsc$w_length] = .desc [dsc$w_length] + .spaces;
    localdesc [dsc$a_pointer] = linebuffer;
!
! Delete trailing spaces
!
    ptr = linebuffer + .localdesc [dsc$w_length];
```

```
; 1572      2280 4         WHILE (
; 1573      2281 4             ptr = .ptr - 1;
; 1574      2282 4             CH$RCHAR (.ptr) EQL %ASCII ' '
; 1575      2283 4             )
; 1576      2284 3             DO localdesc [dsc$w_length] = .localdesc [dsc$w_length] - 1;
; 1577      2285 2     END;
; 1578      2286 2     !
; 1579      2287 2     ! Call caller's routine or LIB$PUT_OUTPUT if caller didn't specify one
; 1580      2288 2     !
; 1581      2289 2     helpinfo [hlp$l_tabindex] = 0;
; 1582      2290 2     IF .typeout_routine NEQ 0
; 1583      2291 3         THEN BEGIN
; 1584      2292 3             flags = .helpinfo [hlp$l_hlpflags] AND %X'FFFF';            !Trim flags to user-only flags
; 1585      2293 3             RETURN (.typeout_routine) (localdesc, flags, user_data, helpinfo [hlp$l_curlevel]);
; 1586      2294 3             END
; 1587      2295 2         ELSE RETURN lib$put_output (localdesc)
; 1588      2296 2
; 1589      2297 1     END;                                    ! Of call_output


                                      0FFC 00000 CALL_OUTPUT:
                                                     .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11     ; 2224
                      5E     FF50   CE  9E 00002     MOVAB   -176(SP), SP                             ;
                      58       04   AC  D0 00007     MOVL    HELPDATA, R8                             ; 2253
                      59       04   A8  D0 0000B     MOVL    4(R8), R9                                ;
                               10   A8  D5 0000F     TSTL    16(R8)                                   ; 2256
                                    06  13 00012     BEQL    1$                                       ;
                      5B       10   A8  D0 00014     MOVL    16(R8), R11                              ; 2257
                                    06  11 00018     BRB     2$                                       ;
                      50            6E  9E 0001A 1$: MOVAB   A_ZERO, R0                               ; 2256
                      5B            50  D0 0001D     MOVL    R0, R11                                  ;
                                    6E  D4 00020 2$: CLRL    A_ZERO                                   ; 2264
  08        00        6E            00  2C 00022     MOVC5   #0, (SP), #0, #8, LOCALDESC              ; 2265
                               F8   AD     00027                                                      ;
                      57       08   AC  D0 00029     MOVL    DESC, R7                                 ; 2266
                                    67  B5 0002D     TSTW    (R7)                                     ;
                                    4B  13 0002F     BEQL    7$                                       ;
                               04   A7  D5 00031     TSTL    4(R7)                                    ; 2267
                                    46  13 00034     BEQL    7$                                       ;
            05        69            01  E0 00036     BBS     #1, (R9), 3$                             ; 2269
                               0C   A8  D5 0003A     TSTL    12(R8)                                   ;
                                    04  13 0003D     BEQL    4$                                       ;
                                    56  D4 0003F 3$: CLRL    SPACES                                   ; 2270
                                    0B  11 00041     BRB     5$                                       ;
                      50       14   A9  D0 00043 4$: MOVL    20(R9), R0                               ; 2271
            56        50            01  78 00047     ASHL    #1, R0, SPACES                           ;
                      56            02  C0 0004B     ADDL2   #2, SPACES                               ;
  56        20        6E            00  2C 0004E 5$: MOVC5   #0, (SP), #32, SPACES, LINEBUFFER        ; 2272
                               08   AE     00053                                                      ;
                      5A            53  D0 00055     MOVL    R3, PTR                                  ;
            6A    04  B7            67  28 00058     MOVC3   (R7), @4(R7), (PTR)                      ; 2273
     F8     AD        67            56  A1 0005D     ADDW3   SPACES, (R7), LOCALDESC                  ; 2274
              FC      AD       08   AE  9E 00062     MOVAB   LINEBUFFER, LOCALDESC+4                  ; 2275
                      50       08   AE  9E 00067     MOVAB   LINEBUFFER, R0                           ; 2279
```

```
                          5A      F8  AD  3C 0006B          MOVZWL  LOCALDESC, PTR                                  ;
                          5A          50  C0 0006F          ADDL2   R0, PTR                                         ;
                          20          7A  91 00072 6$:      CMPB    -(PTR), #32                                     ; 2282
                                      05  12 00075          BNEQ    7$                                              ;
                                  F8  AD  B7 00077          DECW    LOCALDESC                                       ; 2284
                                      F6  11 0007A          BRB     6$                                              ;
                                  1C  A9  D4 0007C 7$:      CLRL    28(R9)                                          ; 2289
                                  0C  A8  D5 0007F          TSTL    12(R8)                                          ; 2290
                                      14  13 00082          BEQL    8$                                              ;
                          04      AE      69 3C 00084       MOVZWL  (R9), FLAGS                                     ; 2292
                                  14  A9  9F 00088          PUSHAB  20(R9)                                          ; 2293
                                      5B  DD 0008B          PUSHL   R11                                             ;
                                  0C  AE  9F 0008D          PUSHAB  FLAGS                                           ;
                                  F8  AD  9F 00090          PUSHAB  LOCALDESC                                       ;
                          0C      B8      04 FB 00093       CALLS   #4, @12(R8)                                     ;
                                          04 00097          RET                                                     ; 2295
                                  F8  AD  9F 00098 8$:      PUSHAB  LOCALDESC                                       ;
                  00000000G       00      01 FB 0009B       CALLS   #1, LIB$PUT_OUTPUT                              ;
                                          04 000A2          RET                                                     ; 2297

; Routine Size:  163 bytes,    Routine Base:  $CODE$ + 0D06
```

```
;  1591          2298  1 %SBTTL  'Routine is_key_on_line';
;  1592          2299  1 ROUTINE is_key_on_line (helpinfo, linedesc, level, keydesc) =
;  1593          2300  2 BEGIN
;  1594          2301  2 !++
;  1595          2302  2 ! This routine scans the line described by linedesc to see if
;  1596          2303  2 ! it is a keyword line or a qualifier line.
;  1597          2304  2 !
;  1598          2305  2 ! Inputs:
;  1599          2306  2 !
;  1600          2307  2 !       helpinfo        Address of help info vector (pointed to by help data vector)
;  1601          2308  2 !       linedesc        Address of string descriptor for the line
;  1602          2309  2 !
;  1603          2310  2 ! Outputs:
;  1604          2311  2 !
;  1605          2312  2 !       level           level found is returned
;  1606          2313  2 !       keydesc         filled in with string descriptor for found key/qualifier
;  1607          2314  2 !
;  1608          2315  2 ! Return values:
;  1609          2316  2 !
;  1610          2317  2 !       true            key/qualifier found, level and keydesc filled in
;  1611          2318  2 !       false           not a key/qualifier line
;  1612          2319  2 !
;  1613          2320  2 !--
;  1614          2321  2
;  1615          2322  2 MAP
;  1616          2323  2     helpinfo : REF BBLOCK,
;  1617          2324  2     linedesc : REF BBLOCK,
;  1618          2325  2     keydesc  : REF BBLOCK;
;  1619          2326  2
;  1620          2327  2 LOCAL
;  1621          2328  2     lineptr,
;  1622          2329  2     curchar;
;  1623          2330  2
;  1624          2331  2 helpinfo [hlp$v_qualine] = false;                !Not a qualifier line
;  1625          2332  2 helpinfo [hlp$v_keyline] = false;                ! or a key line
;  1626          2333  2 IF .linedesc [dsc$w_length] EQL 0                 !If 0-length line
;  1627          2334  2     THEN RETURN false;                            ! there can be no key on line
;  1628          2335  2 lineptr = .linedesc [dsc$a_pointer];
;  1629          2336  2 curchar = CH$RCHAR (.lineptr);
;  1630          2337  3 IF (.curchar LEQU %ASCII'0'                       !If not numeric
;  1631          2338  3     OR .curchar GTRU %ASCII'9')
;  1632          2339  2     AND .curchar NEQ %ASCII'/'                    !And not a qualifier line
;  1633          2340  2 THEN RETURN false                                ! then its not a keyword line
;  1634          2341  3 ELSE BEGIN
;  1635          2342  3     IF .curchar NEQ %ASCII '/'                    !Unless a keyword
;  1636          2343  4         THEN BEGIN
;  1637          2344  4         lineptr = .lineptr - 1;                   !Back up the pointer
;  1638          2345  4         IF NOT skip_blanks (.linedesc, lineptr)   ! and skip blanks
;  1639          2346  4             THEN RETURN false;                    !  and if went to end of line, not special line
;  1640          2347  4         keydesc [dsc$a_pointer] = .lineptr;       !Set pointer to start of key
;  1641          2348  4         keydesc [dsc$w_length] = scan_word (.linedesc, lineptr);
;  1642          2349  4         IF NOT lib$cvt_dtb (.keydesc [dsc$w_length],
;  1643          2350  4                             .keydesc [dsc$a_pointer], .level)
;  1644          2351  4                 THEN RETURN false;
;  1645          2352  4         IF NOT skip_blanks (.linedesc, lineptr) !Skip blanks following key level
;  1646          2353  4                 THEN RETURN false;                !and give up if end of line
;  1647          2354  4         helpinfo [hlp$v_keyline] = true;          !Flag a key line
```

```
; 1648    2355 4          END
; 1649    2356 4          ELSE BEGIN
; 1650    2357 4              helpinfo [hlp$v_qualine] = true;     !'/' -- flag qualifier line
; 1651    2358 3              END;
; 1652    2359 3      keydesc [dsc$a_pointer] = .lineptr;          !Set pointer to keyword or qualifier
; 1653    2360 3      keydesc [dsc$w_length] =  scan_word (.linedesc, lineptr);
; 1654    2361 3      RETURN true;
; 1655    2362 2      END;
; 1656    2363 1  END;                                  ! Of is_key_on_line


                                  001C 00000 IS_KEY_ON_LINE:
                                                     .WORD   Save R2,R3,R4                    ; 2299
                5E            04  C2 00002           SUBL2   #4, SP                           ;
                53      04    AC  7D 00005           MOVQ    HELPINFO, R3                     ; 2331
          03    A3            0C  8A 00009           BICB2   #12, 3(R3)                       ; 2332
                              64  B5 0000D           TSTW    (R4)                             ; 2333
                              7F  13 0000F           BEQL    5$                               ;
                6E      04    A4  D0 00011           MOVL    4(R4), LINEPTR                   ; 2335
                50      00    BE  9A 00015           MOVZBL  @LINEPTR, CURCHAR                ; 2336
                30            50  D1 00019           CMPL    CURCHAR, #48                     ; 2337
                              05  1B 0001C           BLEQU   1$                               ;
                39            50  D1 0001E           CMPL    CURCHAR, #57                     ; 2338
                              05  1B 00021           BLEQU   2$                               ;
                2F            50  D1 00023 1$:       CMPL    CURCHAR, #47                     ; 2339
                              68  12 00026           BNEQ    5$                               ;
                2F            50  D1 00028 2$:       CMPL    CURCHAR, #47                     ; 2342
                              47  13 0002B           BEQL    3$                               ;
                              6E  D7 0002D           DECL    LINEPTR                          ; 2344
                      4010    8F  BB 0002F           PUSHR   #^M<R4,SP>                       ; 2345
       0000V    CF            02  FB 00033           CALLS   #2, SKIP_BLANKS                  ;
                55            50  E9 00038           BLBC    R0, 5$                           ;
                52      10    AC  D0 0003B           MOVL    KEYDESC, R2                      ; 2347
          04    A2            6E  D0 0003F           MOVL    LINEPTR, 4(R2)                   ;
                      4010    8F  BB 00043           PUSHR   #^M<R4,SP>                       ; 2348
       0000V    CF            02  FB 00047           CALLS   #2, SCAN_WORD                    ;
                62            50  B0 0004C           MOVW    R0, (R2)                         ;
                        0C    AC  DD 0004F           PUSHL   LEVEL                            ; 2350
                        04    A2  DD 00052           PUSHL   4(R2)                            ;
                7E            62  3C 00055           MOVZWL  (R2), -(SP)                      ; 2349
   00000000G    00            03  FB 00058           CALLS   #3, LIB$CVT_DTB                  ;
                2E            50  E9 0005F           BLBC    R0, 5$                           ;
                      4010    8F  BB 00062           PUSHR   #^M<R4,SP>                       ; 2352
       0000V    CF            02  FB 00066           CALLS   #2, SKIP_BLANKS                  ;
                22            50  E9 0006B           BLBC    R0, 5$                           ;
          03    A3            04  88 0006E           BISB2   #4, 3(R3)                        ; 2354
                              04  11 00072           BRB     4$                               ; 2342
          03    A3            08  88 00074 3$:       BISB2   #8, 3(R3)                        ; 2357
                53      10    AC  D0 00078 4$:       MOVL    KEYDESC, R3                      ; 2359
          04    A3            6E  D0 0007C           MOVL    LINEPTR, 4(R3)                   ;
                      4010    8F  BB 00080           PUSHR   #^M<R4,SP>                       ; 2360
       0000V    CF            02  FB 00084           CALLS   #2, SCAN_WORD                    ;
                63            50  B0 00089           MOVW    R0, (R3)                         ;
                50            01  D0 0008C           MOVL    #1, R0                           ; 2361
```

```
                 04 0008F        RET                                   ; 2341
            50   D4 00090 5$:    CLRL    R0                            ; 2363
                 04 00092        RET                                   ;

; Routine Size:  147 bytes,    Routine Base:  $CODE$ + 0DA9
```

```
1 %SBTTL  'Routine make_upper_case';
1 ROUTINE make_upper_case (idesc, odesc) =
2 BEGIN
2 !++
2 ! Upper case the name described by string descriptor idesc
2 ! Put the name at location oname
2 !
2 ! Inputs:
2 !
2 !       idesc           Address of string descriptor for input string
2 !
2 ! Outputs:
2 !
2 !       odesc           String descriptor size filled in with right size
2 !                       buffer pointed to by address is uppercased input string
2 !
2 !--
2
2 MAP
2     idesc : REF BBLOCK,
2     odesc : REF BBLOCK;
2 BIND
2     oname = .odesc [dsc$a_pointer] : VECTOR [,BYTE],
2     namlen = idesc[dsc$w_length] : WORD,
2     iname = .idesc[dsc$a_pointer] : VECTOR[,BYTE];
2
2 IF .namlen GTRU 0
2 THEN INCRU i FROM 0 TO .namlen-1
2 DO IF .iname[.i] GEQU %ASCII'a'                       !copy name and convert to upper case
2    AND .iname[.i] LEQU %ASCII'z'
3    THEN oname[.i] = .iname[.i] - (%ASCII'a' - %ASCII'A')
2    ELSE IF .iname [.i] EQL %ASCII ' '                 !If character is space or tab
2              OR .iname [.i] EQL %ASCII '	'
2              OR .iname [.i] EQL 0
3         THEN BEGIN
3              odesc [dsc$w_length] = .i;
3              RETURN true
3              END
2    ELSE oname[.i] = .iname[.i];
2
2 odesc [dsc$w_length] = .namlen;
2 RETURN true
2
1 END;                                                  !Of make_upper_case
```

```
                        001C 00000 MAKE_UPPER_CASE:
                                   .WORD   Save R2,R3,R4                  ; 2365
  51      08  AC  D0 00002         MOVL    ODESC, R1                      ; 2386
  53      04  AC  D0 00006         MOVL    IDESC, R3                      ; 2387
              63  B5 0000A         TSTW    (R3)                           ; 2390
              41  13 0000C         BEQL    7$                             ;
  54          63  3C 0000E         MOVZWL  (R3), R4                       ; 2391
              54  D7 00011         DECL    R4                             ;
```

```
                                              50  D4 00013          CLRL    I                                          ;
                                              33  11 00015          BRB     6$                                         ;
                         52          04 B340  9A 00017 1$:          MOVZBL  @4(R3)[I], R2                              ; 2392
                    61   8F               52  91 0001C              CMPB    R2, #97                                    ;
                                              0E  1F 00020          BLSSU   2$                                         ;
                    7A   8F               52  91 00022              CMPB    R2, #122                                   ; 2393
                                              08  1A 00026          BGTRU   2$                                         ;
         04 B140         52               20  83 00028              SUBB3   #32, R2, @4(R1)[I]                         ; 2394
                                              18  11 0002E          BRB     5$                                         ;
                         20               52  91 00030 2$:          CMPB    R2, #32                                    ; 2395
                                              09  13 00033          BEQL    3$                                         ;
                         09               52  91 00035              CMPB    R2, #9                                     ; 2396
                                              04  13 00038          BEQL    3$                                         ;
                                              52  D5 0003A          TSTL    R2                                         ; 2397
                                              05  12 0003C          BNEQ    4$                                         ;
                         61               50  B0 0003E 3$:          MOVW    I, (R1)                                    ; 2399
                                              0F  11 00041          BRB     8$                                         ; 2400
                    04 B140               52  90 00043 4$:          MOVB    R2, @4(R1)[I]                              ; 2402
                                              50  D6 00048 5$:          INCL    I                                          ; 2392
                         54               50  D1 0004A 6$:          CMPL    I, R4                                      ;
                                              C8  1B 0004D          BLEQU   1$                                         ;
                         61               63  B0 0004F 7$:          MOVW    (R3), (R1)                                 ; 2404
                         50               01  D0 00052 8$:          MOVL    #1, R0                                     ; 2405
                                                  04 00055          RET                                                ; 2407

; Routine Size:  86 bytes,    Routine Base:  $CODE$ + 0E3C
```

```
;  1703          2408  1 %SBTTL  'Routine scan_word';
;  1704          2409  1 ROUTINE scan_word (linedesc, lineptr) =
;  1705          2410  2 BEGIN
;  1706          2411  2 !++
;  1707          2412  2 ! This routine returns the length of the word which is pointed to
;  1708          2413  2 ! by lineptr in the line linedesc describes.  It also advances
;  1709          2414  2 ! lineptr to the character past the end of the word.
;  1710          2415  2 !
;  1711          2416  2 ! Inputs:
;  1712          2417  2 !
;  1713          2418  2 !       linedesc        Address of string descriptor for line
;  1714          2419  2 !       lineptr         Points to beginning of word
;  1715          2420  2 !
;  1716          2421  2 ! Outputs:
;  1717          2422  2 !
;  1718          2423  2 !       lineptr         updated
;  1719          2424  2 !
;  1720          2425  2 ! Return value:
;  1721          2426  2 !
;  1722          2427  2 !       Length of word found
;  1723          2428  2 !
;  1724          2429  2 !--
;  1725          2430  2
;  1726          2431  2 MAP
;  1727          2432  2     linedesc : REF BBLOCK;
;  1728          2433  2
;  1729          2434  2 OWN
;  1730          2435  2     symbolics : VECTOR [96, BYTE] INITIAL
;  1731          2436  2 ('''#$%&''()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|}~');
;  1732          2437  2
;  1733          2438  2 LOCAL
;  1734          2439  2     firstchar,
;  1735          2440  2     ownptr,
;  1736          2441  2     endptr,
;  1737          2442  2     startptr,
;  1738          2443  2     curchar : BYTE;
;  1739          2444  2
;  1740          2445  2 IF .linedesc [dsc$w_length] EQL 0        !If 0-length line
;  1741          2446  2     THEN RETURN 0;                      ! then no word to return
;  1742          2447  2 ownptr = ..lineptr;                     !Get pointer to start of word
;  1743          2448  2 startptr = .ownptr;                     !Remember where it starts
;  1744          2449  2 endptr = .linedesc [dsc$w_length] + .linedesc [dsc$a_pointer]; !Figure end of word
;  1745          2450  2 curchar = CH$RCHAR (.startptr);         ! Get the first character and
;  1746          2451  2 IF CH$FAIL (CH$FIND_CH (93, symbolics, (%X'7F' AND .curchar)))  ! check validity.
;  1747          2452  2     THEN RETURN 0;
;  1748          2453  2 WHILE CH$DIFF (.endptr, .ownptr) GTR 0  !While there is line left
;  1749          2454  3 DO BEGIN
;  1750          2455  3     curchar = CH$A_RCHAR (ownptr);      !Get the character
;  1751          2456  3     IF CH$FAIL (CH$FIND_CH (93, symbolics, (%X'7F' AND .curchar)))
;  1752          2457  3     THEN EXITLOOP;
;  1753          2458  2     END;
;  1754          2459  2 .lineptr = .ownptr;                     ! Return updated pointer
;  1755          2460  2 RETURN .ownptr - .startptr;             !
;  1756          2461  1 END;                                    ! Of scan_word
```

```
                                                        00E92                   .BLKB   2
30  2F  2E  2D  2C  2B  2A  29  28  27  26  25  24  23  22  00E94 SYMBOLICS:
                                                                                .ASCII  \'"#$%&'()**,-./0123456789:;<=>?@ABCDEFGHI\
3F  3E  3D  3C  3B  3A  39  38  37  36  35  34  33  32  31  00EA3
                    49  48  47  46  45  44  43  42  41  40  00EB2
58  57  56  55  54  53  52  51  50  4F  4E  4D  4C  4B  4A  00EBC               .ASCII  \JKLMNOPQRSTUVWXYZ[\<92>\]^_`abcdefghijk\
67  66  65  64  63  62  61  60  5F  5E  5D  5C  5B  5A  59  00ECB
                                        6B  6A  69  68  00EDA
7A  79  78  77  76  75  74  73  72  71  70  6F  6E  6D  6C  00EDE               .ASCII  \lmnopqrstuvwxyz{|}~\<0><0><0>
                                    00  00  00  7E  7D  7C  7B  00EED


                                                    003C 00000 SCAN_WORD:
                                                                                .WORD   Save R2,R3,R4,R5                        ; 2409
                                50          04      AC  D0 00002                MOVL    LINEDESC, R0                            ; 2445
                                                    60  B5 00006                TSTW    (R0)
                                                    50  13 00008                BEQL    5$
                                52          08      BC  D0 0000A                MOVL    @LINEPTR, OWNPTR                        ; 2447
                                54                  52  D0 0000E                MOVL    OWNPTR, STARTPTR                        ; 2448
                                55                  60  3C 00011                MOVZWL  (R0), ENDPTR                            ; 2449
                                55          04      A0  C0 00014                ADDL2   4(R0), ENDPTR
                                53                  64  90 00018                MOVB    (STARTPTR), CURCHAR                     ; 2450
                    50                  53      07  00  EF 0001B                EXTZV   #0, #7, CURCHAR, R0                     ; 2451
                            FF78    CF      005D    8F  50  3A 00020            LOCC    R0, #93, SYMBOLICS
                                                    02  12 00028                BNEQ    1$
                                                    51  D4 0002A                CLRL    R1
                                                    51  D5 0002C 1$:            TSTL    R1
                                                    2A  13 0002E                BEQL    5$
                                52                  55  D1 00030 2$:            CMPL    ENDPTR, OWNPTR                          ; 2453
                                                    1A  15 00033                BLEQ    4$
                                                    52  D6 00035                INCL    OWNPTR                                  ; 2455
                                53                  62  90 00037                MOVB    (OWNPTR), CURCHAR
                    50                  53      07  00  EF 0003A                EXTZV   #0, #7, CURCHAR, R0                     ; 2456
                            FF59    CF      005D    8F  50  3A 0003F            LOCC    R0, #93, SYMBOLICS
                                                    02  12 00047                BNEQ    3$
                                                    51  D4 00049                CLRL    R1
                                                    51  D5 0004B 3$:            TSTL    R1
                                                    E1  12 0004D                BNEQ    2$
                                        08      BC  52  D0 0004F 4$:            MOVL    OWNPTR, @LINEPTR                        ; 2459
                                                52  C2 00053                   SUBL2   STARTPTR, R2                            ; 2460
                                                50  52  D0 00056               MOVL    R2, R0
                                                    04  00059                   RET
                                                50  D4 0005A 5$:               CLRL    R0                                      ; 2461
                                                    04  0005C                   RET

; Routine Size:  93 bytes,    Routine Base:  $CODE$ + 0EF4
```

```
; 1758      2462  1 %SBTTL 'Routine expand_it';
; 1759      2463  1 ROUTINE expand_it ( record_desc ) =
; 1760      2464  2 BEGIN
; 1761      2465  2 !++
; 1762      2466  2 !
; 1763      2467  2 !  This routine provides a common section of code to use if the
; 1764      2468  2 !       help library record is DCX data reduced.
; 1765      2469  2 !
; 1766      2470  2 !--
; 1767      2471  2 BIND
; 1768      2472  2     context = .lbr$gl_control[lbr$l_ctxptr] :  BBLOCK,
; 1769      2473  2     expand_desc = context[ctx$l_dcxrecdsc]: BBLOCK [dsc$c_s_bln];
; 1770      2474  2
; 1771      2475  2 MAP
; 1772      2476  2     record_desc: REF BBLOCK;
; 1773      2477  2
; 1774      2478  2 if .dcxshr_address eql 0
; 1775      2479  2 then
; 1776      2480  2     perform (lbr$load_dcx());
; 1777      2481  2
; 1778      2482  2 expand_desc[dsc$w_length]  = obj$c_maxrecsiz;
; 1779      2483  2 record_desc[dsc$b_dtype]   = dsc$k_dtype_t;
; 1780      2484  2 record_desc[dsc$b_class]   = dsc$k_class_s;
; 1781    P 2485  2 perform((.dcx_expand_data) (context[ctx$l_dcxctx], .record_desc, expand_desc,
; 1782      2486  2                         record_desc[dsc$w_length]));
; 1783      2487  2 record_desc[dsc$a_pointer] = .expand_desc[dsc$a_pointer];
; 1784      2488  2 RETURN true;
; 1785      2489  1 END;


                                 001C 00000 EXPAND_IT:
                                                  .WORD   Save R2,R3,R4                           ; 2463
                50     0000G  CF  D0 00002        MOVL    LBR$GL_CONTROL, R0                      ; 2472
                53        0E  A0  D0 00007        MOVL    14(R0), R3                              ;
                54        5A  A3  9E 0000B        MOVAB   90(R3), R4                              ; 2473
                       0000G  CF  D5 0000F        TSTL    DCXSHR_ADDRESS                          ; 2478
                              08  12 00013        BNEQ    1$                                      ;
        0000G   CF            00  FB 00015        CALLS   #0, LBR$LOAD_DCX                        ; 2480
                26            50  E9 0001A        BLBC    STATUS, 2$                              ;
                64      0800  8F  B0 0001D 1$:    MOVW    #2048, (R4)                             ; 2482
                52        04  AC  D0 00022        MOVL    RECORD_DESC, R2                         ; 2483
           02   A2      010E  8F  B0 00026        MOVW    #270, 2(R2)                             ;
                              52  DD 0002C        PUSHL   R2                                      ; 2486
                              14  BB 0002E        PUSHR   #^M<R2,R4>                              ;
                          52  A3  9F 00030        PUSHAB  82(R3)                                  ;
        0000G   DF            04  FB 00033        CALLS   #4, @DCX_EXPAND_DATA                    ;
                08            50  E9 00038        BLBC    STATUS, 2$                              ;
           04   A2        04  A4  D0 0003B        MOVL    4(R4), 4(R2)                            ; 2487
                50            01  D0 00040        MOVL    #1, R0                                  ; 2488
                                  04 00043 2$:    RET                                             ; 2489

; Routine Size:  68 bytes,    Routine Base:  $CODE$ + 0F51
```

```
;  1787          2490  1 %SBTTL  'Routine skip_blanks';
;  1788          2491  1 ROUTINE skip_blanks (linedesc, lineptr) =
;  1789          2492  2 BEGIN
;  1790          2493  2 !++
;  1791          2494  2 ! This routine skips blanks and tabs in the line.
;  1792          2495  2 ! Returns true if skipped to non-blank, non-tab character
;  1793          2496  2 ! Returns false if skipped to exclamation pointer or end of line.
;  1794          2497  2 !
;  1795          2498  2 ! Inputs:
;  1796          2499  2 !
;  1797          2500  2 !       linedesc        Address of string descriptor for current line
;  1798          2501  2 !       lineptr         Address of pointer to current spot in line
;  1799          2502  2 !
;  1800          2503  2 ! Outputs:
;  1801          2504  2 !
;  1802          2505  2 !       lineptr         updated
;  1803          2506  2 !
;  1804          2507  2 ! Return values:
;  1805          2508  2 !
;  1806          2509  2 !       true            more to come
;  1807          2510  2 !       false           no non-blank, non-tab, non-comment found
;  1808          2511  2 !
;  1809          2512  2 !--
;  1810          2513  2
;  1811          2514  2 MAP
;  1812          2515  2     linedesc : REF BBLOCK;
;  1813          2516  2
;  1814          2517  2 LOCAL
;  1815          2518  2     retval,
;  1816          2519  2     ownptr,
;  1817          2520  2     endptr,
;  1818          2521  2     curchar;
;  1819          2522  2
;  1820          2523  2 IF .linedesc [dsc$w_length] EQL 0        !If 0-length line
;  1821          2524  2     THEN RETURN false;                   ! then end of line
;  1822          2525  2 ownptr = ..lineptr;                      !Make a copy of the pointer
;  1823          2526  2 endptr = .linedesc [dsc$w_length] + .linedesc [dsc$a_pointer] - 1;
;  1824          2527  2 WHILE CH$DIFF (.endptr, .ownptr) GTR 0
;  1825          2528  3 DO BEGIN
;  1826          2529  3     curchar = CH$A_RCHAR (ownptr);
;  1827          2530  3     IF .curchar EQL %ASCII '!'
;  1828          2531  4         THEN BEGIN
;  1829          2532  4              .lineptr = .ownptr;
;  1830          2533  4              RETURN false;
;  1831          2534  3              END;
;  1832          2535  3     IF .curchar NEQ %ASCII ' '
;  1833          2536  3         AND .curchar NEQ %ASCII '       '
;  1834          2537  4         THEN BEGIN
;  1835          2538  4              .lineptr = .ownptr;
;  1836          2539  4              RETURN true;
;  1837          2540  3              END;
;  1838          2541  2     END;
;  1839          2542  2 .lineptr = .ownptr;
;  1840          2543  2 RETURN false;                           !Went to end of line
;  1841          2544  1 END;                                    !Of skip_blanks
```

```
                                          0004 00000 SKIP_BLANKS:
                                                          .WORD    Save R2                                    ; 2491
                      50       04     AC  D0 00002        MOVL     LINEDESC, R0                               ; 2523
                                      60  B5 00006        TSTW     (R0)                                       ;
                                      33  13 00008        BEQL     3$                                         ;
                      52       08     BC  D0 0000A        MOVL     @LINEPTR, OWNPTR                           ; 2525
                      51              60  3C 0000E        MOVZWL   (R0), R1                                   ; 2526
             50       51       04     A0  C1 00011        ADDL3    4(R0), R1, R0                              ;
                                      50  D7 00016        DECL     ENDPTR                                     ;
                      52              50  D1 00018 1$:    CMPL     ENDPTR, OWNPTR                             ; 2527
                                      1C  15 0001B        BLEQ     2$                                         ;
                                      52  D6 0001D        INCL     OWNPTR                                     ; 2529
                      51              62  9A 0001F        MOVZBL   (OWNPTR), CURCHAR                          ;
                      21              51  D1 00022        CMPL     CURCHAR, #33                               ; 2530
                                      12  13 00025        BEQL     2$                                         ;
                      20              51  D1 00027        CMPL     CURCHAR, #32                               ; 2535
                                      EC  13 0002A        BEQL     1$                                         ;
                      09              51  D1 0002C        CMPL     CURCHAR, #9                                ; 2536
                                      E7  13 0002F        BEQL     1$                                         ;
                 08   BC              52  D0 00031        MOVL     OWNPTR, @LINEPTR                           ; 2538
                      50              01  D0 00035        MOVL     #1, R0                                     ; 2539
                                          04 00038        RET                                                 ;
                 08   BC              52  D0 00039 2$:    MOVL     OWNPTR, @LINEPTR                           ; 2542
                                      50  D4 0003D 3$:    CLRL     R0                                         ; 2544
                                          04 0003F        RET                                                 ;

; Routine Size:  64 bytes,    Routine Base:  $CODE$ + 0F95


; 1842            2545  1 END                                   ! Of module
; 1843            2546  0 ELUDOM




;                                  PSECT SUMMARY
;
;       Name                       Bytes                       Attributes
;
;   $CODE$                          4053  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)


;                              Library Statistics
;
;                                          -------- Symbols --------       Pages      Processing
;       File                               Total   Loaded   Percent      Mapped      Time
;
;   _$255$DUA28:[SYSLIB]STARLET.L32;1       9776       17         0         581        00:01.0
```

```
;                                       COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:GETHELP/OBJ=OBJ$:GETHELP MSRC$:GETHELP/UPDATE=(ENH$:GETHELP)

; Size:          3893 code + 160 data bytes
; Run Time:         01:17.4
; Elapsed Time:     03:01.3
; Lines/CPU Min:     1973
; Lexemes/CPU-Min: 23340
; Memory Used:   324 pages
; Compilation Complete
```

GETHELP
LIS

INDEX
LIS

GETPUT
LIS

GETMEM
LIS